Advant® OCS

AdvaCommand® Runtime
AdvaCommand® Development
AdvaCommand® for Windows

User's Guide

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

Advant, AdvaCommand, AdvaInform, and AdvaBuild are registered trademarks of ABB Asea Brown Boveri Ltd, Switzerland.

AdvaSoft is a trademark of ABB, Asea Brown Boveri.

SQL*Net is a registered trademark of Oracle Corporation.

Microsoft, MS, MS-DOS are registered trademarks and Windows is a trademark of Microsoft Corporation.

Intellution and SPECIFIX are registered trademarks, and THE FIX, FIX MMI, FIX DMACS, FIX BOS, FIX*ation*, and Click and Slick are trademarks of Intellution Inc.

All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Industrial Systems AB . ABB Industrial Systems AB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Industrial Systems AB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Industrial Systems AB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Industrial Systems AB ’s written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

CE MARKING

This product meets the requirements specified in EMC Directive 89/336/EEC and in Low Voltage Directive 72/23/EEC.

Copyright © ABB Industrial Systems AB 1997.

3BSE001264/D

Template: 3BSE001286/E

TABLE OF CONTENTS

Chapter 1 - Introduction

1.1 General Information 1-1
  1.1.1 Intended Audience 1-2
1.2 Equipment Requirements 1-2
1.3 Manual Organization 1-3
1.4 Conventions 1-4
1.5 Documentation 1-5
  1.5.1 AdvaCommand for Windows Documentation Binders 1-6
    1.5.1.1 AdvaCommand for Windows User's Guides summary ... 1-6
  1.5.2 Communication related documentation 1-8
1.6 Release History 1-8
1.7 Terminology 1-9
  1.7.1 Acronyms 1-9
  1.7.2 Definition of Terms 1-10
1.8 Product Overview 1-11
  1.8.1 AdvaCommand Runtime 1-12
  1.8.2 AdvaCommand Development 1-12
  1.8.3 AdvaCommand with FIX database 1-13
1.9 User Interface 1-14

Chapter 2 - Installation

2.1 Site-Planning Environment 2-1
2.2 Setup 2-1
  2.2.1 Sub-installation configuration 2-3
2.3 Shutdown Procedures 2-3
  2.3.1 Normal shutdown 2-3
  2.3.2 Emergency shutdown 2-3
2.4 Startup Procedures 2-4
2.5 Product Verification 2-4

Chapter 3 - Configuration/Application Building

3.1 Design Considerations 3-1
  3.1.1 Reference to manuals 3-1
  3.1.2 Conditions before starting 3-1
    3.1.2.1 System configuration 3-1
    3.1.2.2 Object referencing 3-2
    3.1.2.3 Configuration of SQL access towards Advant Station 500 IMS 3-2

CONTENTS (continued)

3.1.2.4 Verifying network communication 3-2
3.2 Capacity and Performance 3-2
3.3 Application Startup 3-3
3.4 Tutorial 3-3
3.5 Application Procedures 3-4
3.5.1 Introduction to display construction 3-4
3.5.2 Introduction to display areas and layout 3-6
3.5.2.1 Display element color use 3-6
3.5.3 Predefined templates 3-8
3.5.3.1 Common display properties 3-11
3.5.4 Monitor resolution 3-12
3.5.5 Predefined variables and display properties 3-13
3.5.5.1 Global variables 3-13
3.5.6 Reusable graphic objects, Dynamos 3-18
3.5.7 Dynamo structure 3-19
3.5.7.1 Dynamo related displays 3-21
3.5.7.2 Display element naming 3-22
3.5.7.3 Dynamo sets 3-23
3.5.8 Object attributes reference 3-24
3.5.8.1 Analog Input 3-25
3.5.8.2 Analog Output 3-27
3.5.8.3 Digital Input 3-28
3.5.8.4 Digital Output 3-30
3.5.8.5 DAT 3-31
3.5.9 Object subscription methods 3-33
3.5.9.1 Specifying subscription update time 3-33
3.5.9.2 Specifying individual bits within status words 3-35
3.5.9.3 Creating and using custom Dynamos 3-36
3.5.10 Building custom process displays 3-40
3.5.10.1 Defining display properties 3-40
3.5.11 Display header configuration 3-41
3.5.11.1 Display title field 3-41
3.5.11.2 Date and time field 3-42
3.5.11.3 Alarm line properties 3-42
3.5.12 Defining function keys 3-43
3.5.12.1 Dynamic function keys 3-43
3.5.12.2 Keyboard function keys 3-44

CONTENTS (continued)

3.5.13 Defining object close-up properties 3-44
  3.5.13.1 Create a new Close-up window 3-47
3.5.14 Defining Display menus 3-48
  3.5.14.1 Predefined Display menu 3-48
  3.5.14.2 Adding new groups 3-49
  3.5.14.3 Creating a new group page 3-50
  3.5.14.4 Adding paging keys 3-51
3.5.15 Group displays 3-52
  3.5.15.1 Designing group displays 3-52
3.5.16 Object displays 3-53
  3.5.16.1 Creating a custom object display 3-54
  3.5.16.2 Assigning custom object displays to objects 3-54
3.5.17 Alarm summary 3-55
3.5.18 Creating a custom Alarm Summary 3-55
  3.5.18.1 Custom defined Alarm and event treatment texts 3-56
  3.5.18.2 Parameters used for Alarm and Text handling 3-57
  3.5.18.3 Alarm line fields 3-62
  3.5.18.4 Changing display font size 3-62
  3.5.18.5 Changing column order 3-62
  3.5.18.6 Creating separate lists for alarm areas 3-63
  3.5.18.7 Configuration of the Historical trend data display 3-64
3.6 Configuration/Application Building Menus 3-65

Chapter 4 - Runtime Operation

4.1 Product Operation 4-1
4.2 Operating Overview 4-1
  4.2.1 Display areas and objects 4-2
    4.2.1.1 Screen header 4-3
    4.2.1.2 Display area 4-4
    4.2.1.3 Dialog area 4-5
    4.2.1.4 Object close-up area 4-5
  4.2.2 Predefined display types 4-6
    4.2.2.1 Display menus 4-6
    4.2.2.2 Group displays 4-7
    4.2.2.3 Object displays 4-8
    4.2.2.4 Alarm and event lists 4-9
    4.2.2.5 Trend displays 4-10
4.3 Runtime Tutorial 4-10

CONTENTS (continued)

4.4 Operating Instructions 4-11
4.4.1 Keyboard operation 4-11
4.4.1.1 Fixed keyboard keys 4-11
4.4.1.2 Dynamic keyboard keys 4-12
4.4.2 Operating with a mouse 4-12
4.4.3 Choosing displays 4-12
4.4.3.1 Selection from the display menu 4-13
4.4.3.2 Direct display selection 4-14
4.4.3.3 Previous display selection 4-15
4.4.3.4 Paging 4-16
4.4.4 Monitoring and supervisory control 4-17
4.4.4.1 Selecting a process object 4-18
4.4.4.2 Monitoring a process object 4-20
4.4.4.3 Controlling process objects 4-21
4.4.4.4 Process control from object displays 4-22
4.4.4.5 Controlling analog objects 4-23
4.4.5 Event and alarm handling 4-23
4.4.5.1 Presentation of the last unacknowledged process alarm 4-26
4.4.5.2 Acknowledgment of alarms 4-26
4.5 Runtime Operation Menus 4-27

Chapter 5 - Maintenance

5.1 Preventive Maintenance 5-1
5.2 Hardware Indicators 5-1
5.3 Error Messages 5-1
5.3.1 AdvaSoft FIX interface error messages 5-1
5.4 Fault-Finding and User Repair 5-4
5.4.1 Troubleshooting 5-4
5.4.1.1 Correcting corrupted historical log files 5-4
5.4.2 How to get support from ABB 5-4

Appendix A - References

A.1 ADVASOFT.INI A-1

Appendix B - References

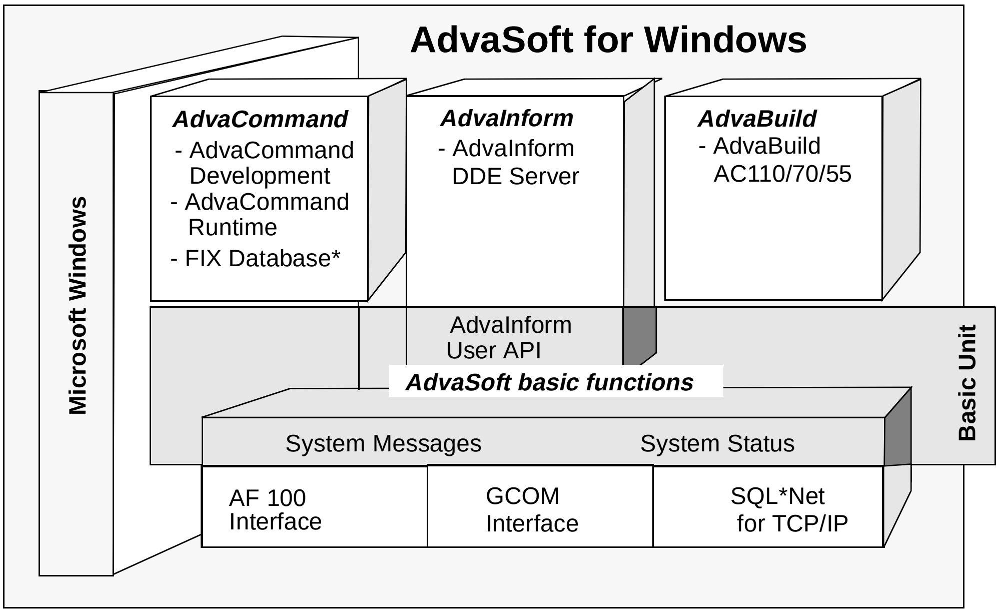
B.1 HTD.INI B-1

INDEX

Chapter 1 Introduction

1.1 General Information

The AdvaCommand for Windows software family provides functionality including process monitoring, supervisory control, alarming and alarm management, real-time and historical trending, distributed networking and comprehensive reporting. It takes advantage of Microsoft Windows' user-friendly interface, coupled with integration to the Advant OCS, to offer predefined template displays, display elements, and functions for process control and supervision.



* Both versions of AdvaCommand, Development and Runtime, are available with the FIX Database option

Figure 1-1. AdvaCommand participation in the AdvaSoft for Windows program suite

To accommodate the different requirements of the **application building** and **plant operation** phases, AdvaCommand for Windows is available in two options, AdvaCommand Runtime and AdvaCommand Development.

AdvaCommand Runtime

AdvaCommand Runtime for plant operation, which provides real-time control or supervision, offers the following functionality:

- Monitoring, to display process data graphically or numerically in real time.
- Supervisory control, to enable the operator to monitor and change process object setpoints interactively.
- Alarming, to provide the ability to recognize and report alarms and events to the operator.
- Historical data collection, trending and reporting, the sampling and storing of data points in the system, to be retrieved to trend displays and operator-defined reports.

AdvaCommand Development

AdvaCommand Development extends AdvaCommand Runtime with graphical tools to draw and define static and dynamic properties for all types of displays and functions.

A library of predefined AdvaCommand displays with static and dynamic properties is included. This is comprised of process and object displays, group displays and alarm lists that are compatible with the displays and objects of the Advant OCS.

Graphical tools included are:

- Dynamos, re-usable graphic wizards for easily defining and saving custom-built objects.
- Command language, for assigning dynamic functions to displays and objects from within the Draw application.
- Macros, for automating operator tasks through a series of commands and instructions.

AdvaCommand with FIX database

Both the runtime and development versions of AdvaCommand are available with the FIX database. AdvaCommand with FIX database includes the FIX DMACS SCADA option. This option makes it possible for you to include the FIX standard options such as communication drivers (I/O drivers) or to use the FIX DataBase Builder (DBB). In AdvaCommand, you can have process displays and other applications with data from the FIX database mixed with data from the Advant OCS system via the Advant Fieldbus 100 Interface or via the GCOM Interface.

1.1.1 Intended Audience

If you are an application, system or maintenance engineer, you can use this guide during application building and maintenance. Plant operators will also find Chapters 1 and 4 useful. This is the primary document in the AdvaCommand for Windows documentation set, and it includes descriptions of procedures and predefined displays.

1.2 Equipment Requirements

Disk space, memory and operating system requirements are explained in this section of the *AdvaSoft for Windows User's Guide*.

1.3 Manual Organization

In this guide you will find information about, and instructions on how to use, AdvaCommand for Windows. Figure 1-2 below shows what each chapter and section in this guide include.

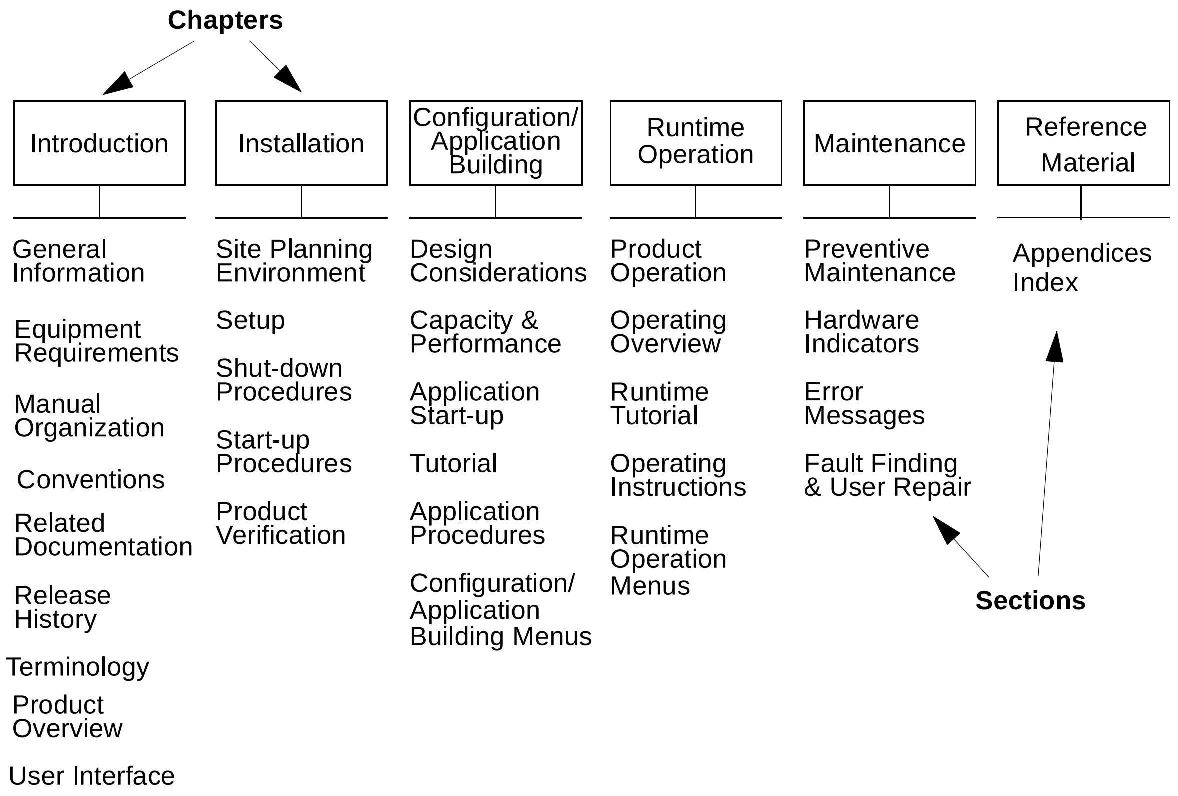


Figure 1-2. Structure of this guide

Chapter 1, Introduction

Includes an overview of the documentation and the AdvaCommand for Windows product.

Chapter 2, Installation

Includes site planning and setup. AdvaCommand for Windows is installed together with the AdvaSoft for Windows product, described in the corresponding user's guide.

Chapter 3, Configuration/Application Building

Includes instructions for system configuration, display design and application building procedures.

Chapter 4, Runtime Operation

Includes runtime operating procedures using predefined displays and objects to monitor and control a process with an Advant OCS.

Chapter 5, Maintenance

Includes preventive maintenance and troubleshooting with explanations of error messages.

1.4 Conventions

This user's guide features consistent visual formats for easy recognition. The following conventions are used throughout this guide:

Examples	**Convention description**
Document names	Document names are in ***italics***
Abbreviations	Names abbreviated are written in full the first time they are presented, followed by the abbreviation. For example "...personal computer (PC)..."
Mouse instructions	"Click" or "Double-click" means pointing with the mouse cursor, then clicking the left button.
Mouse actions	You are instructed to **choose**, which means clicking to execute an action (like buttons and menu items). You are instructed to **select**, which means clicking to set a mark with a selection cursor (like highlighting text or selecting check boxes).
Press single keys	Displayed in **Bold** and starting with an uppercase letter for keys which are labeled on the keyboard, for example, "... press <**Ctrl>**..." or "... press the **Ctrl** key...," or **bold** in lowercase letters for keys that are not labeled, for example "...press <**enter>**..."
Press and release more than one key	Separated by a comma (,), for example, "... press and release **<Alt, F4>**..."
Press more than one key at a time	Separated by a dash (-), for example, "... press **<Ctrl-C>**..."
Screen buttons	Displayed in **BOLD** and uppercase letters, for example,"... press the **OK** button..."
Menus and menu items	Displayed in **Bold** and names of menus or menu items begin with uppercase letters
Program source text	Displayed in `Courier` font

Instructions to type	Characters you are instructed to type are in **bold** File names to enter appear in quotation marks, like "File name"
Performing operations in a sequence	When you are asked to perform sequences of operations, they are numbered in chronological order

1.5 Documentation

Below is an overview of various required and optional documentation available from ABB, depending on your system configuration, followed by a short description of each guide:

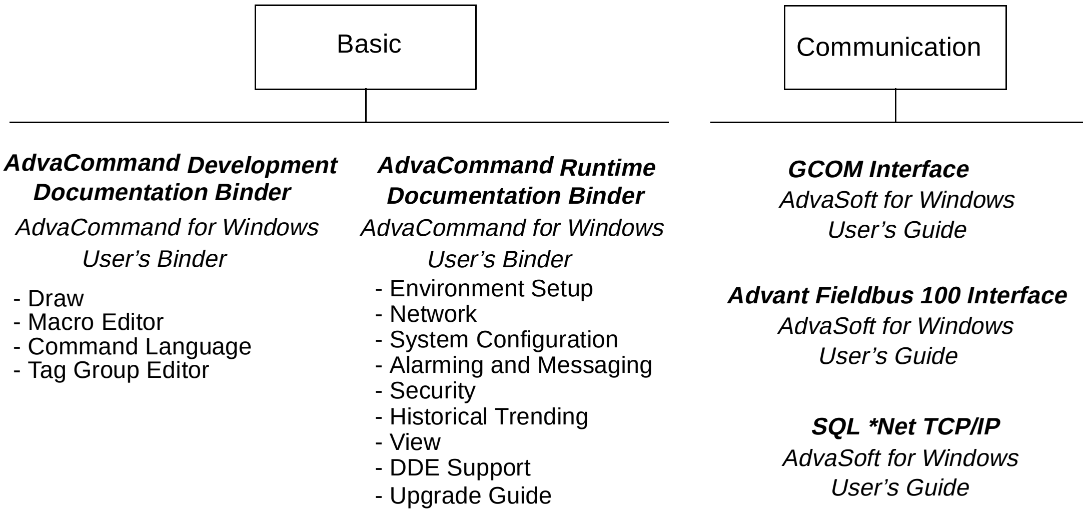

Basic
Communication
AdvaCommand Development Documentation Binder
AdvaCommand for Windows User's Binder
- Draw
- Macro Editor
- Command Language
- Tag Group Editor
AdvaCommand Runtime Documentation Binder
AdvaCommand for Windows User's Binder
- Environment Setup
- Network
- System Configuration
- Alarming and Messaging
- Security
- Historical Trending
- View
- DDE Support
- Upgrade Guide
GCOM Interface
AdvaSoft for Windows User's Guide
Advant Fieldbus 100 Interface
AdvaSoft for Windows User's Guide
SQL *Net TCP/IP
AdvaSoft for Windows User's Guide

Figure 1-3. AdvaCommand for Windows and Related Documentation Tree

1.5.1 AdvaCommand for Windows Documentation Binders

Introduction

This manual is the primary document for the AdvaCommand Runtime and AdvaCommand Development user’s guides. It includes design and runtime **procedures**. The different user’s guides explain in detail **how** to perform these procedures, and there are references to them throughout this guide.

Table 1-1. AdvaCommand Runtime Documentation Binder

System Setup	System Development	Display Development	Advanced tools
Environment Setup Network System Configuration Upgrade Guide	Alarming and Messaging Security	View	Historical Trending DDE Support

Table 1-2. AdvaCommand Development Documentation Binder

System Setup	System Development	Display Development	Advanced tools
		Draw Macro Editor Command Language Tag Group Editor	

In addition there are two manuals dedicated to the FIX Database handling:

- Database Builder/AdvaCommand for Windows, User’s Guide
- Database Reference/AdvaCommand for Windows, User’s Guide.

1.5.1.1 AdvaCommand for Windows User’s Guides summary

Environment Setup

Includes recommendations for selecting hardware and software components. It is the reference manual for tasks explained in Chapters 3, 4 and 5 of this guide.

System Configuration

Explains how to use the System Configuration Utility (SCU). It is the reference manual for tasks explained in Chapters 3 and 4 of this guide.

View

For plant operators, provides instructions about how to use displays developed with the Draw program. It is the reference manual for operations explained in Chapters 3 and 4 of this guide.

Draw

Complete instructions for developing interactive operator displays using Draw. It is the reference manual for tasks explained in Chapter 3 of this guide.

Macro Editor

Guide to designing macros. It is the reference manual for tasks explained in Chapter 3 of this guide.

Command Language

Guide to the command language, used widely to control dynamic properties in Draw. It is the reference manual for tasks explained in Chapter 3 of this guide.

Tag Group Editor

Guide to building tag groups and implementing them into displays. It is the reference manual for tasks explained in Chapter 3 of this guide if tag groups are used.

Historical Trending

Explains how to design and use historical assign, collect and trend displays. It is the reference manual for tasks explained in Chapters 3 and 4 of this guide.

Alarming and Messaging

Guide to implementing alarming and messaging strategy and managing alarms. It is the reference manual for tasks explained in Chapters 3 and 4 of this guide.

Security

Describes how to configure the security system. It is the reference manual for tasks explained in Chapters 3, 4 and 5 of this guide, if used.

DDE Support

Explains how to design and use DDE for report generation. It is the reference manual for tasks explained in Chapters 3 and 4 of this guide.

Database Builder/AdvaCommand for Windows

User’s guide which describes how to design and build process databases with the FIX database.

Database Builder/AdvaCommand for Windows

Reference manual providing detailed information about each FIX database block. Use it in conjunction with the *DataBase Builder User’s Guide*.

1.5.2 Communication related documentation

GCOM
AdvaSoft for Windows
User's Guide

This guide gives you instructions on configuration and the use of the GCOM Interface for AdvaSoft for Windows.

Advant Fieldbus 100 Interface
AdvaSoft for Windows
User's Guide

This guide gives you instructions on the configuration and use of the Advant Fieldbus 100 Interface for AdvaSoft for Windows.

SQL*Net TCP/IP
AdvaSoft for Windows User's Guide

Includes instructions on installation and configuration of the SQL*Net TCP/IP and the Oracle ODBC driver.

1.6 Release History

Each AdvaCommand for Windows option has its own release history, which consists of a version number indicating a major release with new functionality and a revision number indicating minor improvements and error corrections.

Table 1-3. AdvaCommand for Windows Release History

Unit	Version/ Revision	Description
AdvaCommand Runtime	1.0	Initial version FIX version 5.0/b
	1.1	Support for the GCOM protocol FIX version 5.5/30
	1.2	FIX version 5.6
AdvaCommand Development	1.0	Initial version FIX version 5.0/b
	1.1	New enhanced object display elements added for support of the GCOM protocol. FIX version 5.5/30
	1.2	FIX version 5.6

1.7 Terminology

Also refer to the glossaries in the ***Draw*** and ***View*** manuals.

1.7.1 Acronyms

AC
AC stands for Advant Controller, the common term for the Advant Controller series of controllers used in Advant OCS, for example, AC 110, AC 410.

AF 100
AF 100 stands for Advant Fieldbus 100.

AS
AS stands for Advant Station, the common term for the Advant Station series of operator, engineering and information management stations within the Advant OCS, for example, AS 100IMS.

DCS
DCS stands for Distributed Control System, a process station with a distributed database, for example, Advant Controller 410. This is a common term for such systems.

DDE
DDE stands for Dynamic Data Exchange, a form of interprocess communication that uses shared memory to exchange data between applications in the MS-Windows environment, as in DDE Server.

GCOM
GCOM stands for Gateway COMmunication protocol.

GUI
GUI stands for Graphical User Interface.

MB
MB stands for megabytes, as in "...10 MB of disk space...."

MMI
MMI stands for Man Machine Interface, a common term for operator interactive functionality.

MS
MS stands for Microsoft Corporation. All Microsoft programs mentioned are preceded by "MS," as in MS Word.

ODBC
ODBC stands for Open DataBase Connectivity, an interface providing vendor-neutral access to various database systems.

OS
OS stands for Operator Station, a station with operator interactive functionality.

SCADA
SCADA stands for Supervisory Control And Data Acquisition.

SCADA node
A SCADA node is a station that holds a database in memory and collects process information.

SCU
SCU stands for System Configuration Utility for the FIX DMACS.

SQL
SQL stands for Structured Query Language, a standardized language for query of relational databases.

TCP/IP
TCP/IP stands for Transmission Control Protocol/Internet Protocol, a standard communication protocol.

1.7.2 Definition of Terms

Acknowledge
Acknowledge sets a flag to an alarm indicating that the alarm has been observed.

Advant OCS
Refers to one or more of the distributed control systems from ABB Industrial Systems, for example, ABB Master and ABB Mod 300.

Attribute (object)
Attribute (object) is additional information included in a process or calculated object, for example, the process value.

Bar graph
A bar graph is a display object which fills or empties according to the process object to which it is linked.

Cyclic (update of data)
Cyclic (update of data) refers to a task being executed or a value being updated within certain time intervals, for example, cyclic update of an analog input, with a cycle time of 1 s.

Display element
Display elements are the dynamic building blocks of displays. A display element is a view of one or more process or system objects, for example, an analog input signal, a pump or a valve. It provides information about the objects and defines the interaction between them and the user. A display element can be as simple as a numerical value or as complex as a complete object display for a process object.

Display object
A display object is a defined object contained in a display, either graphic or text, which can contain dynamic properties.

Dynamic Keys
Dynamic keys are used in runtime in the process dialogs to operate the objects that are selected in a display. They are also used to call up displays or start applications. Dynamic keys can be activated from either the keyboard or the screen.

Dynamo
An object, link, or group of objects that are stored in a Dynamo set (re-usable graphic wizards).

Event (update of data)
When a value or object is updated, causing a change to occur, or an alarm limit is exceeded, data is event-updated.

Field (object attribute)
See attribute.

Node
A node is a network unit, with its own address, communicating on a network. A node can be an AdvaSoft for Windows station, a gateway or an Advant Controller.

Process Dialog
A process dialog is a set of executable operations that can be performed in runtime at a process object. Process dialogs are performed via dynamic keys.

Process object
A process object is a physical or calculated process object, or a functional unit containing all related inputs or outputs. For example, an analog input including value, limits, status, etc.

Real-time
Values updated from a process as they happen are called real-time.

Subscription (to objects and fields)
Subscription (to objects and fields) is subscription to data, i.e., accessing input data from an automation system.

Tag (object)
A process or calculated object.

1.8 Product Overview

To accommodate the different requirements of the **application building** and **plant operation** phases, AdvaCommand for Windows is available in two options:

- **AdvaCommand Runtime**, consisting of software to control and supervise a plant via a color-graphic MS-Windows-style user interface.
- **AdvaCommand Development**, which includes tools to build applications using predefined template displays.

The figure below shows the relationship between parts of the AdvaCommand for Windows product, followed by a more detailed summary of the functionality of the product.

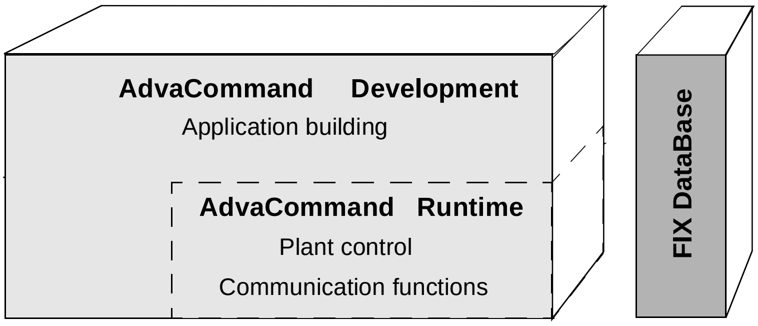


Figure 1-4. AdvaCommand for Windows product overview

1.8.1 AdvaCommand Runtime

AdvaCommand Runtime presents real-time data on displays developed with AdvaCommand Development to monitor, supervise and control, and list and report real-time and historical process data. AdvaCommand Runtime is available independent of, and runs independent of, AdvaCommand Development. It includes the following functionality:

- **View** is the primary program for monitoring the ability to display real-time process data to operators in text, numerically and in graphical formats and colors.
- **Alarming** includes lists from the whole plant, the automation system and/or dedicated process section lists. The operator is always notified of unacknowledged process and system alarms. The operator can acknowledge and choose to look at the alarm lists from any display from the display header area.
- **User-Based Security** enables the operator to precisely control all access to data and system applications. Users can be assigned both individual and group rights, and audit trails let the operator track and record operator actions.
- **Real-Time and Historical Trending** enables collection, storing and display of real-time process data at rates specified by the operator. The data can be presented in multi-pen trend, multi-bar trend, XY plot and histogram for analysis and optimization.
- **Reporting via DDE** generates graphs and reports through spreadsheets or word processors such as Microsoft Excel or Word. The Excel macros that are provided feature easy-to-use dialog boxes which make report setup easy. The Report Generator macros allow access to process data and historical data, create real-time charts and schedule reports for automatic generation.
- **Networking** enables sharing of the distributed client/server architecture, data and alarms across the network.
- **SQL support** allows the operator to collect real-time process data through SQL commands using Command Language Scripts in View or to retrieve data that is stored in a relational database accessible through the AdvaInform SQL*Connect and to display the data in a Historical Display chart.

1.8.2 AdvaCommand Development

AdvaCommand Development includes AdvaCommand Runtime plus the tools to design applications using predefined template displays and functions for efficient engineering.

AdvaCommand Development includes graphical design tools for designing all aspects of the runtime displays and functions, as summarized below.

- **Draw** is the primary design program and includes an intuitive graphical user interface, configurable tool box, color box and dialog boxes. The display objects and properties include text, rectangle, and oval and polygon objects with foreground color, background color, edge color, pattern style and edge style properties.

  The **dynamic properties** for displays and display objects include foreground color, background color, edge coloring, horizontal and vertical movement, scale, rotate, and horizontal and vertical fill.

The **Quick View** function enables quick switching to runtime mode for efficient testing of designed displays and functions.

Draw uses the full capabilities of the supporting engineering-optimization functions:

- **Dynamos** are graphic display elements which provide higher level re-usable graphic presentation.
- **Tag group editor** is used to create generic pictures that are identical except for the tagnames they reference.
- **Command language** is a scripting tool to automate operator tasks through a series of instructions. These instructions are then executed from View.
- **Macros editor** is used to assign simple keystroke sequences to macros built with the Command Language.
- **Predefined and modifiable templates and displays**, including AdvaCommand process object displays (AI, AO, DI and DO) and dynamic function keys, all include predefined dynamic properties using Dynamos for efficient engineering. Display types are the basic display, display menu, alarm lists, large and close-up object displays, and group display.

1.8.3 AdvaCommand with FIX database

Both the runtime and development versions of AdvaCommand are available with the FIX database. AdvaCommand with FIX database includes the FIX DMACS SCADA option. This option makes it possible for you to include the FIX standard options such as communication drivers (I/O drivers) or to use the FIX DataBase Builder (DBB).

In AdvaCommand, you can have process displays and other applications with data from the FIX database mixed with data from the Advant OCS system via the Advant Fieldbus 100 Interface or via the GCOM Interface.

NOTE

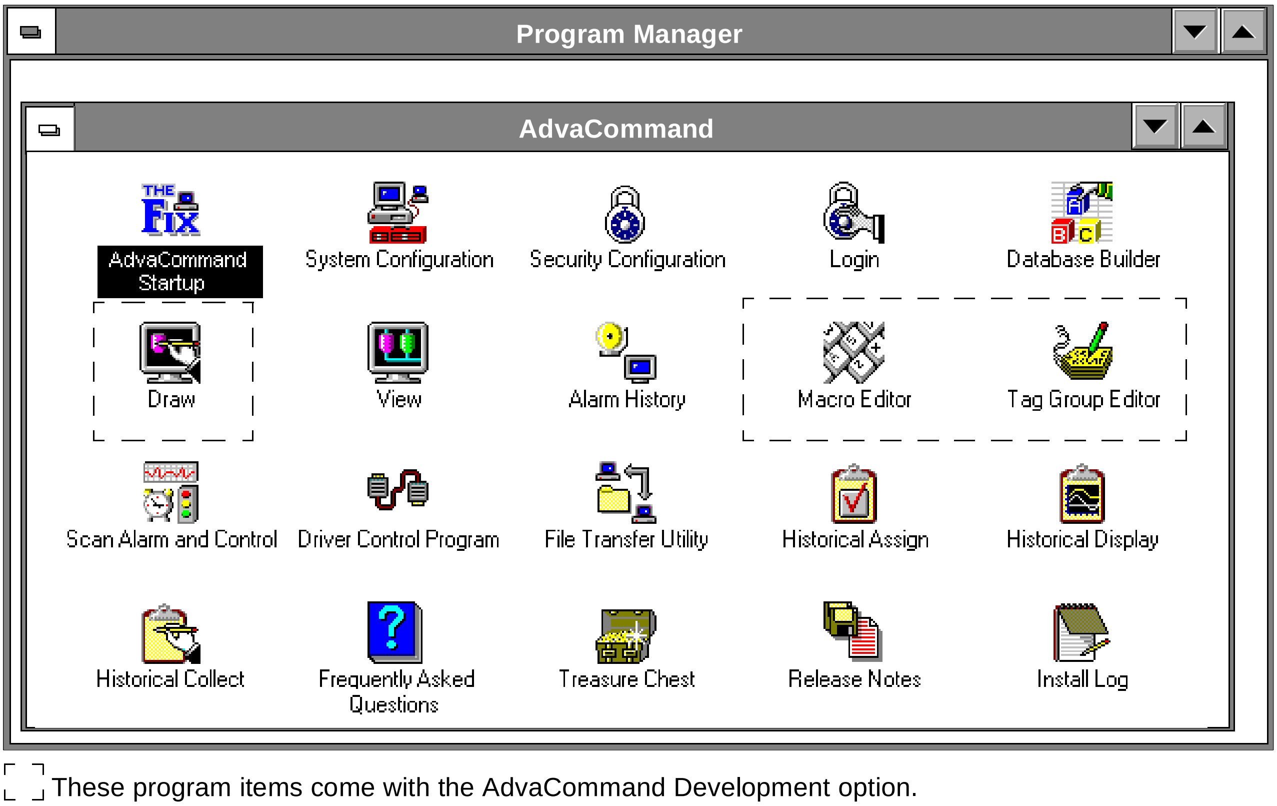
You cannot use the predefined dynamos for the Advant OCS process objects for presentation of data from the FIX database.

You access data from the Advant Fieldbus 100 network and from the GCOM network from AdvaCommand via the AdvaInform User API.
Advant Fieldbus 100 and GCOM data are not directly available in the FIX database. To operate with Advant Fieldbus 100 or GCOM data, you must build a separate application (that uses FIX DDE Server or FIX Easy Database Access (EDA)) that copies the data to the FIX database.

1.9 User Interface

The AdvaCommand for Windows user interface is based on standard MS-Windows and the AdvaCommand program group.

Figure 1-5 shows the user interfaces included in AdvaCommand Development.



These program items come with the AdvaCommand Development option.

Figure 1-5. AdvaCommand for Windows User Interface

Chapter 2 Installation

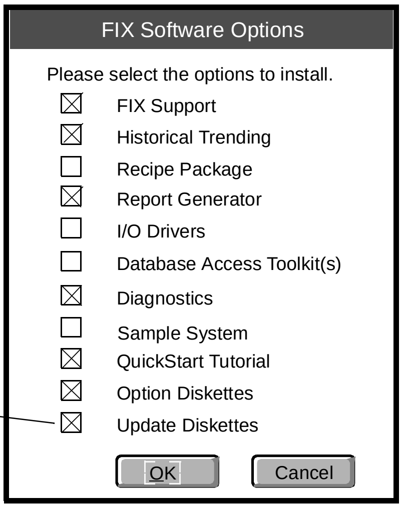
2.1 Site-Planning Environment

Site planning pertains to business- or plant-specific environmental, earthing and power supply requirements and classification. Refer to the ***AdvaSoft for Windows User’s Guide*** and other relevant hardware documentation to make sure you are in compliance with these requirements.

2.2 Setup

From the AdvaSoft for Windows installation program, you choose to install the AdvaCommand Development or AdvaCommand Runtime option. Follow the procedures and insert diskettes as prompted.

1. Select **Custom Install**.
2. Select the following options from the AdvaCommand installation window described below.



Please select the options to install.

☒ FIX Support
☒ Historical Trending
☐ Recipe Package
☒ Report Generator
☐ I/O Drivers
☐ Database Access Toolkit(s)
☒ Diagnostics
☐ Sample System
☒ QuickStart Tutorial
☒ Option Diskettes
☒ Update Diskettes

This option must be marked to have FIX 5.6 functionality

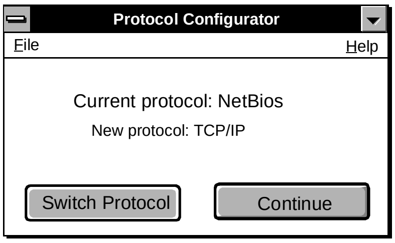
Figure 2-1. Customized installation window

NOTE

To obtain the correct AdvaSoft installation settings, edit the default selections in accordance with Figure 2-1 above.

3. Give a unique node name to your AdvaSoft station.
4. When you are prompted to set up System Configuration Utility (SCU) information, such as SCADA, Network, etc., click the **SKIP** button.
5. Select communication protocol between FIX nodes.

   During installation, the following dialog *may* be displayed if you have both Microsoft TCP/IP and NetBios installed on your computer.



Current protocol: NetBios
New protocol: TCP/IP

Figure 2-2. Protocol Configurator display

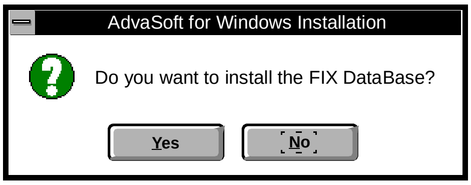
   Protocol choice only affects communication between pure FIX nodes. The protocols chosen do not affect the AdvaCommand application and the AdvaSoft protocols used.

6. Choose **EXIT Setup**.

   You are now returned to AdvaSoft's main installation program.

   A message box, which asks for the WDMACS directory location, is displayed. Click the OK button for the default location.

7. A dialog is shown which asks if you want to use the FIX's application database builder.



Do you want to install the FIX DataBase?

Figure 2-3. FIX Database dialog

   You can only use the FIX Database when the appropriate hardware key is connected to the PC terminal. The hardware key is supplied with the FIX Database option.

   When you finish this step, the AdvaCommand for Windows installation is completed. Before you start the application for the first time, however, you must do some configuration.

2.2.1 Sub-installation configuration

1. Before your first startup of AdvaCommand for Windows, open the SCU application.

   Open the **Settings** menu

   - Select **Tasks**
     Remove the **WSACTASK** from the startup list
     Select **Paths**
     Add the path to the AdvaCommand displays **…\PIC\ABBPIC**.

2. Modify the HTC startup

   - Select the Historical Collect icon in the AdvaCommand program group.
   - Open the Windows Program Manager **File** menu.
   - Choose **Properties**.
   - Append /S on the command line, after HTC.EXE, to suppress a messagebox reporting lost communication.

You can set up View to suppress the message box '*Could not write to this field*'. The message box is shown if you try to select a previously selected object or if a command fails for any other reason. Do the following to suppress the error message:

1. Select the View icon in the AdvaCommand program group.
2. Open the Windows Program Manager **File** menu.
3. Choose **Properties**.
4. Change the command line to VIEW.EXE -S1203.

If View automatically starts up, make command line changes in the SCU as well.

1. Start System Configuration from the AdvaCommand program group.
2. Choose Configure and Tasks....
3. Change the command line to VIEW.EXE -S1203.

2.3 Shutdown Procedures

2.3.1 Normal shutdown

Shut down AdvaCommand for Windows and associated software as follows:

- Choose the **Shutdown AdvaCommand** item from the **Options** menu. To shut down associated communication option(s), see the corresponding user's guide(s).

2.3.2 Emergency shutdown

If an application error occurs and a program halts, exit all running programs and Microsoft Windows, if possible, by choosing the **Exit** item in the **File** menu in each program. Then end your Microsoft Windows session in the same way. Restart Microsoft Windows from MS-DOS, then restart each unit in the right order from the Microsoft Windows desktop.

2.4 Startup Procedures

For all AdvaCommand options:

> Double-click on the **AdvaCommand startup** icon, and an automatic start of associated software, including the communication option, is executed.

2.5 Product Verification

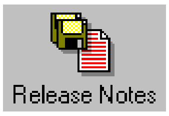


Figure 2-4. AdvaCommand release notes

For verification and information about your AdvaCommand for Windows option, double-click the **Release Notes** icon from the AdvaCommand program group.

Chapter 3 Configuration/Application Building

3.1 Design Considerations

3.1.1 Reference to manuals

As the various phases of application building are described in this chapter, there are references to the AdvaCommand for Windows documentation set listed in the table below. We recommend you use this guide together with the appropriate manual for the current design phase.

Also refer to Section 1.5.1, AdvaCommand for Windows Documentation Binders for an overview of the manuals and their contents.

Table 3-1. Using AdvaCommand for Windows documentation for engineering tasks

Design Phase	Manual
Calculating capacity and performance.	Environment Setup
Designing displays including dynamic and static properties, alarming.	Draw
Tag groups are not used in predefined displays. You may want to assign tag groups to custom process and object displays.	Tag Group Editor
Assigning functions to keystrokes, screen buttons, screens and objects. Command language instructions are included in predefined displays.	Command Language Key Macros
Designing alarming and messaging strategy. Not necessary to use when standard predefined strategy is used.	Alarming and Messaging
Designing user accounts, user access and write protection.	Security
Testing displays and data exchange.	View
Performance, notes on additional components and certain configurations.	Environment Setup
Designing and building a FIX database	Database Builder Database Reference

3.1.2 Conditions before starting

3.1.2.1 System configuration

When you install AdvaCommand for Windows, a default configuration is presumed and stored. The node name is specified at installation time. To change the default configuration, use the AdvaCommand for Windows **System Configuration** program. Double-click on the node name on the monitor displayed in the SCU window.

3.1.2.2 Object referencing

The object database, including tagnames and attributes (unit, process section, description) entered with the Advant Fieldbus 100 Interface configurator, is presumed to be complete at this stage. Procedures in this chapter require knowledge of existing object names and attributes.

When you use the GCOM Interface communication option, no object database is needed.

3.1.2.3 Configuration of SQL access towards Advant Station 500 IMS

Setup Database Source

Verify that the Oracle ODBC driver is installed. If it is not, see installation information in the ***SQL*Net TCP/IP AdvaSoft for Windows User's Guide***.

Setup Database Source in the SCU

Define the Database Source as a SQL Login Account in the SCU.

1. Select **Configure SQL** in the SCU.
2. Select **"Add"**.
3. Write in the following data as SQL login information:

   DB type: **Oracle**
   User Name:**ims_user**
   Password: See *Advant Station 500 IMS User's Guide*
   Database identifier: The name of AS 500IMS, configured during ODBC driver setup.

Also refer to the ***Command Language*** manual.

3.1.2.4 Verifying network communication

To test dynamic data exchange with other stations over a network, the nodes (Advant Controllers or Stations) and communication interface(s) must be configured and running.

3.2 Capacity and Performance

Refer to the ***Environment*** manual.

The capacity of AdvaCommand for Windows depends on the PC, use of third-party applications and communication interfaces. Refer to their respective manuals.

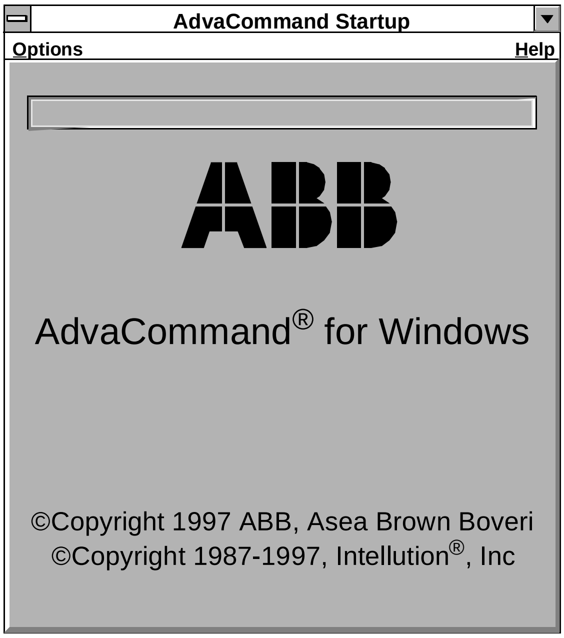
For more information about performance and capacity, see the Release Notes.

3.3 Application Startup

AdvaCommand
Startup

Start AdvaCommand for Windows by double-clicking on the **AdvaCommand start-up** icon.

This starts AdvaCommand for Windows, AdvaSoft for Windows and included ABB communication software, if installed.



AdvaCommand® for Windows

©Copyright 1997 ABB, Asea Brown Boveri
©Copyright 1987-1997, Intellution®, Inc

Figure 3-1. AdvaCommand main window

3.4 Tutorial

Refer to the QuickStart Tutorial in the AdvaCommand Program Group.

3.5 Application Procedures

This section guides you through designing applications and includes references to the AdvaCommand documentation set. The predefined displays and elements mentioned in this chapter are supplied with AdvaCommand for Windows.

3.5.1 Introduction to display construction

When you construct an application using the Draw application, you actually construct different displays and then link them together to form your system. You can use the templates and display elements supplied with AdvaCommand for easier and quicker application development.

If desired, you can construct your system using only custom-built displays. If you do so, however, you lose the system conformity and ease of adoption to ABB Controllers which you have when you use the AdvaCommand templates and display elements. Each template or display element has its intended use, for example, to show information specific for a process area or for the object type currently monitored.

Below are some examples of displays where templates intended for Process displays, Display menus and Group displays have been used, see Figure 3-2.

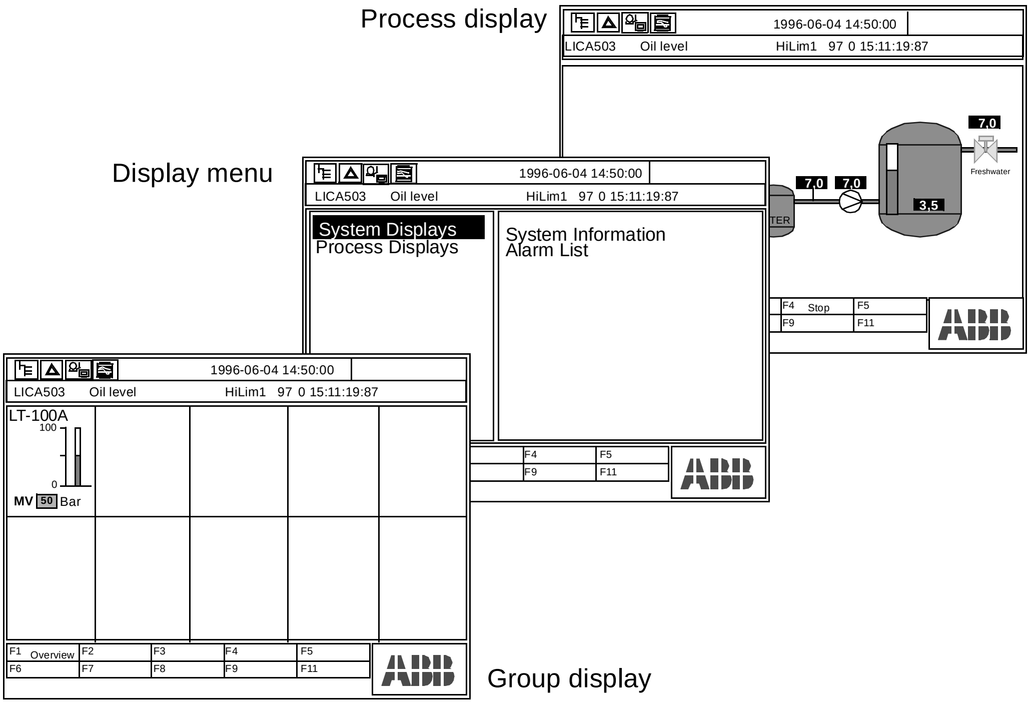


Figure 3-2. Display examples

Before you begin to develop individual displays, consider designing a hierarchy. A hierarchy is simply a system of linked displays that provide progressively more specific information as you move through the displays. When you use the various AdvaCommand templates, choose those most suitable for the intended use. For example, use the **_DMPAGE** template to construct a menu display.

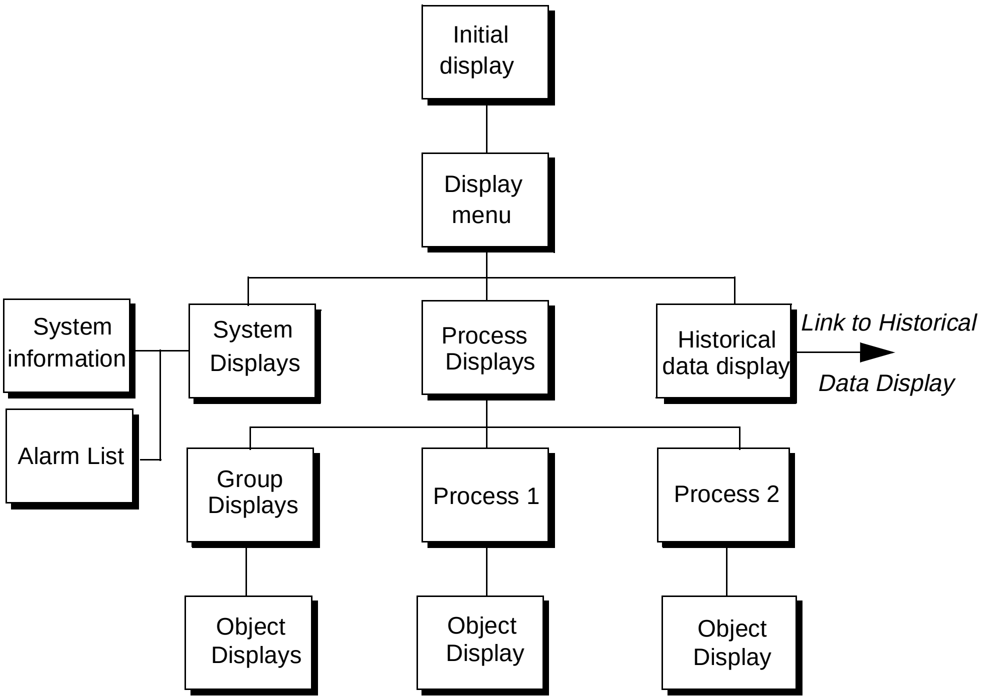


Figure 3-3. Example of a hierarchy of displays

When you use AdvaCommand's templates, there is an existing relationship between the different displays. For example, every display is equipped with a previous button in the dialog button display. Additionally, you can navigate through the system via a **Display Menu**, an **Alarm History**, an **Alarm list** and a **Historical Display Icon** in the **Screen Header**, which provide direct access to the respective displays. See Section 3.5.2, Introduction to display areas and layout.

By providing users with a way to navigate through your system, they can quickly get to the data they require. You can choose to make moving from display to display either flexible in an operator controlled environment or predefined in a secured environment.

3.5.2 Introduction to display areas and layout

Figure 3-4 below shows the standard display layout, featuring predefined areas where you can build displays and function keys.

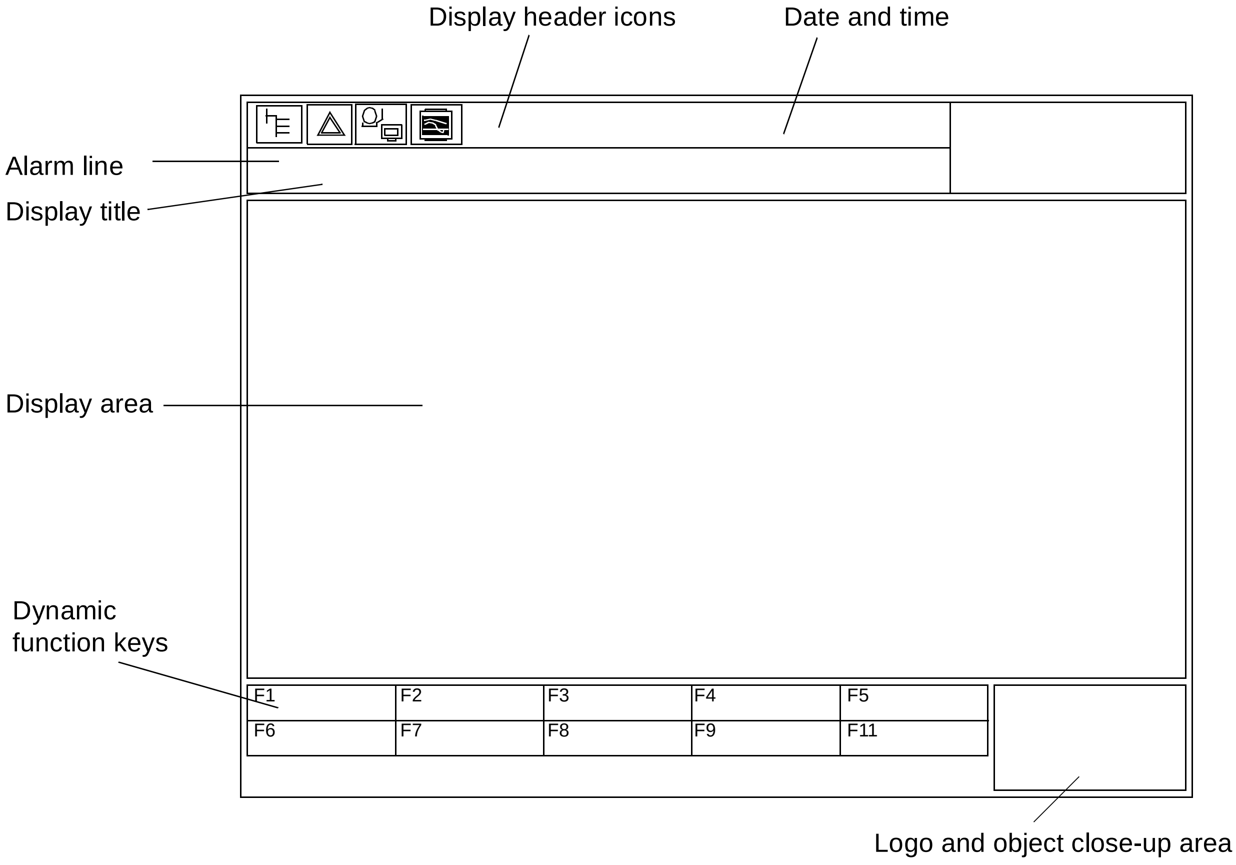


Figure 3-4. Display areas

3.5.2.1 Display element color use

You decide how a process signal or a process object is to be presented in a display by selecting a suitable dynamo from those supplied by AdvaCommand or by creating a custom display element.

The dynamo or display element shows the status of the signals and objects using symbols and color coding. Independent of the display element type, some predefined colors are recommended to represent the status in the table below.

The colors are categorized in order of priority. The highest priority is listed first, that is, flashing red, which stands for an unacknowledged alarm.

Table 3-2. Status colors

Color	Description
Flashing red	Unacknowledged alarm
Yellow	Blocked Input/Output/Commands
Red	Acknowledged alarm, in alarm state, or Process/Signal error

.

Table 3-3. Status indicator colors

Color	Description
White frame	Selected
Yellow B or dotted line	Alarm blocked
Yellow P	Print blocked

.

Table 3-4. Type colors

Color	Description
Green	Default and measured value
Cyan	DO and AO in Manual mode
Magenta	DO and AO in Auto mode

.

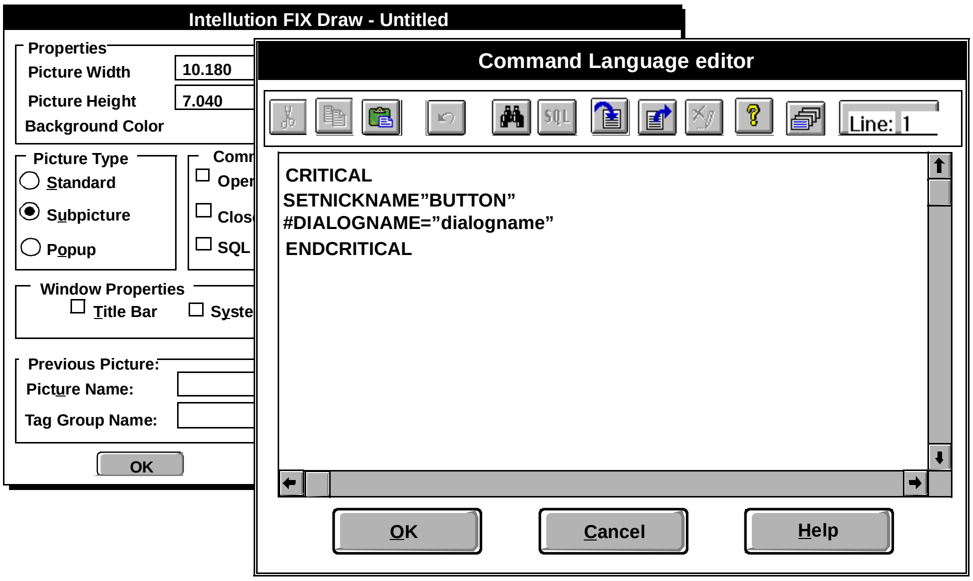
Table 3-5. Static colors

Color	Description
Grey	Support information, e. g., labels
Dark grey	Background

3.5.3 Predefined templates

You can get a head start designing displays with the predefined templates that AdvaCommand provides. Using predefined templates guarantees uniformity throughout all the system and process displays.

When you use an AdvaCommand predefined template to create a display, the Draw application saves the picture as a normal operator display (.ODF) file and does not overwrite the template file. Predefined template properties such as background color, command scripts, area size and placement, and dialog buttons remain constant for a basic, uniform appearance and functionality. You can examine the properties in the **Picture** dialog, which is displayed when you select **Picture...** from the **Edit** menu, see Figure 3-5.



Picture Width 10.180
Picture Height 7.040

Picture Type
○ Standard
● Subpicture
○ Popup

☐ Oper
☐ Clos
☐ SQL

☐ Title Bar ☐ Syste

Picture Name:
Tag Group Name:

Line: 1

```
CRITICAL
SETNICKNAME”BUTTON”
#DIALOGNAME=”dialogname”
ENDCRITICAL
```

Figure 3-5. Properties dialog

The tables below list the predefined template files supplied with AdvaCommand for Windows. They are, together with the predefined dynamos, the building blocks for you to design your own custom process and object displays, used together with your application specific library of type circuits in the Advant Controller(s).

There are **templates** for the basic displays. Select from the list by choosing the **New** item from the **File** menu of the program. The **basic displays** include static and dynamic properties such as predefined standard variables and functions.

NOTE

The file names listed below are preceded by “_”. This distinguishes them from your custom displays. If you save your custom displays without the underscore, they will appear alphabetically in the list after the predefined displays.

The **object displays** include static and dynamic properties for each basic object. Each object has an object display and a close-up display.

Table 3-6. Templates and dynamic function key templates

Name	Description	Area
_DISPDLG	Dialog including Display menu and previous display button	Dynamic function keys
_DMPAGE	Display menu	Display menu group
_GRNPGN	Display menu group display	Display
_GRPDSP	Basic group display	Display
_OBJPDLG	Dynamic function keys for basic display	Dynamic function keys
_PROCDSP	Basic process display	Display

AdvaCommand Development includes a predefined process display with instructions how to access data via SQL. The display name is SQL_samp. The SQL_samp display is set up as a process display in the default Display menu, see Section 3.5.14, Defining Display menus.

Table 3-7. Predefined displays and dynamic function keys

Name	Description	Area
_ALLIST	Alarm list display	Alarm display
_ASUMDLG	Alarm list summary display dialog	Dynamic function keys
_DSPMDLG	Display menu dialog	Dynamic function keys
_DSPMENU	Display menu	Display menu
_GR1PG1	Group display including small object display example	Display
_INFODLG	About information display dialog	Dynamic function keys
_INFOSCR	About information display	Display
_LOGO	ABB logo	Object close-up
_MAIN	Basic display area including display nicknames and positioning information	Display
_TOPLINE	Display header including alarm line and icons	Display header

Table 3-8. Predefined object and object close-up displays and predefined function keys

Name	Description	Area
_AIOBDLG	AI object dynamic function keys	Dynamic function keys
_AIOBJ	AI object display	Display
_AIPLDLG	AI process display dynamic function keys	Dynamic function keys
_AIWI00	Initialization of variables used in close-up and object displays	Object close-up
_AIWI01	AI close-up display	Object close-up
_AOOBDLG	AO object dynamic function keys	Dynamic function keys
_AOOBJ	AO object display	Display
_AOPLDLG	AO process display dynamic function keys	Dynamic function keys
_AOWI00	Initialization of variables used in close-up and object displays	Object close-up
_AOWI01	AO close-up display	Object close-up
_DIOBDLG	DI object dynamic function keys	Dynamic function keys
_DIOBJ	DI object display	Display
_DIPLDLG	DI process display dynamic function keys	Dynamic function keys
_DIWI00	Initialization of variables used in close-up and object displays	Object close-up
_DIWI01	DI close-up display	Object close-up
_DOOBDLG	DO object dynamic function keys	Dynamic function keys
_DOOBJ	DO object display	Display
_DOPLDLG	DO process display dynamic function keys	Dynamic function keys
_DOWI00	Initialization of variables used in close-up and object displays	Object close-up
_DOWI01	DO close-up display	Object close-up

3.5.3.1 Common display properties

The location (and size) of each display type is also specified in the _MAIN.odf display file.

Syntax: OPENPIC { picture | string_var | tag group [X,Y,H,W] } { [nickname|string_var|tag group] [tag group filetag group] }

The predefined displays include the **picture**, **location** and **nickname** argument (except the topline display, which is never replaced and only includes picture and location). The nicknames are used in each display type's commands upon opening.

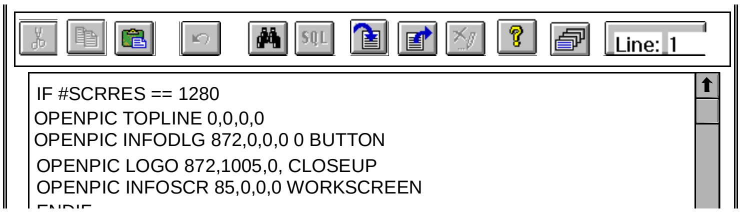
Syntax example: OPENPIC _TOPLINE 4, 4, 0, 0

Table 3-9. Open picture commands in _MAIN.ODF

Display	Command	Comments
TOPLINE	4, 4, 0, 0	"0, 0" means no size specified
INFODLG	872, 4, 0, 0 BUTTON	BUTTON is the nickname for the initial display dialog
LOGO	872, 1005, 0, 0 CLOSEUP	CLOSEUP is the nickname for the logo display for the close-up area
INFOSCR	85, 4, 0, 0 WORKSCREEN	WORKSCREEN is the nickname for the initial display

3.5.4 Monitor resolution

This section tells you how to create displays for monitors with various resolutions. When you use the AdvaCommand templates, monitor resolution is not an issue. A command script in the **_MAIN.ODF**, found in the **...\WDMACS\PIC\ABBPIC** directory, ensures that the template resolution is set to the current monitor resolution, assuming it is one of the resolutions listed below.



```
IF #SCRRES == 1280
OPENPIC TOPLINE 0,0,0,0
OPENPIC INFODLG 872,0,0,0 0 BUTTON
OPENPIC LOGO 872,1005,0, CLOSEUP
OPENPIC INFOSCR 85,0,0,0 WORKSCREEN
ENDIF
```

Figure 3-6. Command Language Editor dialog

TOPLINE.xxx, **INFODLG.xxx**, **LOGO.xxx**, **GR1PG1.xxx** and **INFOSCR.xxx** are dummy displays used to rescale the real displays (**_TOPLINE.odf**, **_INFODLG.odf**, **_LOGO.odf**, **_GR1PG1.odf** and **_INFOSCR.odf**).

The only displays you must resize are the dummy displays. The resolution automatically changes when you replace the displays if the **fit-to-window** parameter, found in the **VIEW.INI** file, is set to **TRUE**. The **VIEW.INI** file is located in the **'...\WDMACS\LOCAL**' directory .

The dummy displays, found in **WDMACS\PIC\ABBPIC**, exist for the following resolutions (with associated process display sizes):

Table 3-10. Display resolutions

Dummy display names	Available Resolutions	Process Display size
*.128	1280 x 1024	12,74x7,82
*.102	1024 x 768	10,18x5,84
*.080	800 x 600	7,94x4,44
*.064	640 x 480	6,31x3,645

To avoid problems with character scaling, etc., when you are creating the process displays, build the displays with the display size for the current resolution. During AdvaCommand installation, the default resolution is set to the resolution currently used on the PC where AdvaCommand is being installed.

NOTE

- It is not necessary to edit the _MAIN.ODF file with regard to display resolution.
- If the picture is too big to save in **Edit** Picture..., enable the title bar checkbox.

3.5.5 Predefined variables and display properties

The **_MAIN.ODF** display file, together with the **TOPLINE.ODF** display file, contains all global variables and common display properties initiated when the runtime operation is started with **View**. **TOPLINE.ODF** also contains some common variables.

3.5.5.1 Global variables

The defined variables cover all the predefined displays included with AdvaCommand. To add or change variables, choose **Commands on opening** from the **Edit Picture** dialog box in **Draw**. The table below describes all the predefined variables used.

Syntax example: DECLARE #NAME STRING GLOBAL

A complete description of the command language syntax is included in the ***Command Language*** manual.

The variables found in the table below are used by the standard displays and dynamos and corresponding command language scripts.

Table 3-11. Variable declarations in _MAIN.ODF

Variable	Type and Scope	Description
#DNAME	STRING GLOBAL	Current display name, globally available
#DTITLE	STRING GLOBAL	Process and object display title, displayed in topline
#LASTDISPLAY	STRING GLOBAL	Previous display, for selecting from the dynamic function key
#LASTDIALOG	STRING GLOBAL	Previous dynamic function key file name
#LASTGROUP	STRING GLOBAL	Previous displayed group page in Display menu
#DIALOGNAME	STRING GLOBAL	Current dynamic function key file name
#TAGNAME	STRING GLOBAL	Tagname of current object
#A_NAME	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the tagname.
#A_DESCR	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the signal description.

Table 3-11. Variable declarations in _MAIN.ODF (Continued)

Variable	Type and Scope	Description
#A_UNIT	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the engineering unit tag.
#F_VALUE	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the value.
#F_R_MIN	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the min. engineering unit.
#F_R_MAX	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the max. engineering unit.
#NAME	STRING GLOBAL	Variable holding the value retrieved by A_NAME
#DESCR	STRING GLOBAL	Variable containing the description retrieved by A_DESCR
#UNIT	STRING GLOBAL	Variable containing the engineering unit tag retrieved by A_UNIT
#R_MAX	STRING GLOBAL	Variable containing the max. range retrieved by F_R_MAX
#R_MIN	STRING GLOBAL	Variable containing the min. range retrieved by F_R_MIN
#BL	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the signal's alarm state.
#SCRRES	NUMERIC GLOBAL	Holds the value of #GS_SCREEN_WIDTH
#STATUS_LINE	STRING GLOBAL	Keeps the “Node.Tag selected by another user” text string.
#STATUS_LINE_VISIBLE	NUMERIC GLOBAL	Set to 0 = status line not visible Set to 1 = status line visible

.

Table 3-12. Variable declarations in _TOPLINE.ODF

Variable	Type and Scope	Description
#ALARM	NUMERIC GLOBAL	Object in alarm state
#GRINDEX	NUMERIC GLOBAL	System variable used in Display menu list

Table 3-12. Variable declarations in _TOPLINE.ODF (Continued)

Variable	Type and Scope	Description
#ITEMS_IN_GR_LIST	NUMERIC GLOBAL	Indicates how many menu items there are in group list (on menu display).
#TA	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the signal's alarm state.
#AL	STRING GLOBAL	Temporary string holding a signal's alarm state. Used in object and close-up pictures.
#A_SUB	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the current process section.
#SUBSYST	STRING GLOBAL	Holds the value retrieved by the #A.SUB variable.
#LEN	NUMERIC GLOBAL	Holds the number of characters in the #TAGNAME variable.
#F_HI_LIM1	STRING GLOBAL	Retrieves the High Limit 1 value for AI objects. Used in object and close-up pictures.
#F_HI_LIM2	STRING GLOBAL	Retrieves the High Limit 2 value for AI objects. Used in object and close-up pictures.
#F_LO_LIM1	STRING GLOBAL	Retrieves the Low Limit 1 value for AI objects. Used in object and close-up pictures.
#F_LO_LIM2	STRING GLOBAL	Retrieves the Low Limit 2value for AI objects. Used in object and close-up pictures.
#F_USE_HI_LIM1	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to set the display mode of the High Limit 1 value. Only used for AI objects.
#F_USE_HI_LIM2	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to set the display mode of the High Limit 2 value. Only used for AI objects.
#F_USE_LO_LIM1	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to set the display mode of the Low Limit 1 value. Only used for AI objects.
#F_USE_LO_LIM2	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to set the display mode of the Low Limit 2 value. Only used for AI objects.

Table 3-12. Variable declarations in _TOPLINE.ODF (Continued)

Variable	Type and Scope	Description
#USE_HI_LIM1	NUMERIC GLOBAL	Holds the value retrieved by #F_USE_HI_LIM1. Only used for AI objects.
#USE_HI_LIM2	NUMERIC GLOBAL	Holds the value retrieved by #F_USE_HI_LIM2. Only used for AI objects.
#USE_LO_LIM1	NUMERIC GLOBAL	Holds the value retrieved by #F_USE_LO_LIM1. Only used for AI objects.
#USE_LO_LIM2	NUMERIC GLOBAL	Holds the value retrieved by #F_USE_LO_LIM2. Only used for AI objects.
#STATUS	NUMERIC GLOBAL	Not used.
#SELECTED	NUMERIC GLOBAL	Not used.
#A_SELECTED	STRING GLOBAL	Not used.
#A_CLASS	STRING GLOBAL	Variable holding the concatenation of #TAGNAME and fieldname. Used in object and close-up picture to retrieve the Priority Class.
#CLASS	STRING GLOBAL	Holds the value retrieved by the #A_CLASS variable.
#F_MAX_LIM	STRING GLOBAL	Retrieves the Max Limit value for AO objects. Used in object and close-up pictures.
#F_MIN_LIM	STRING GLOBAL	Retrieves the Min Limit value for AO objects. Used in object and close-up pictures.
#AI_STATUS_BITS[32][1]	TABLE GLOBAL	Array of status bits
#AO_STATUS_BITS[24][1]	TABLE GLOBAL	Array of status bits
#DI_STATUS_BITS[32][1]	TABLE GLOBAL	Array of status bits
#DO_STATUS_BITS[32][1]	TABLE GLOBAL	Array of status bits
#OBJECT_DISPLAY	NUMERIC GLOBAL	Holds name of object display
#ILLNODETAG	STRING GLOBAL	Keeps the "Illegal 'NODE:TAD' - Check selected dynamo" string.

If applied on the hierachial model, you would use the templates as follows, see Figure 3-7.

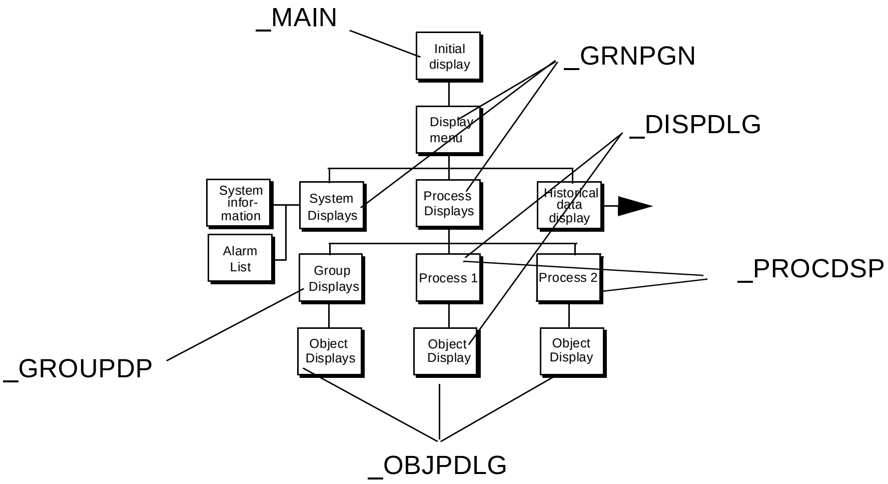
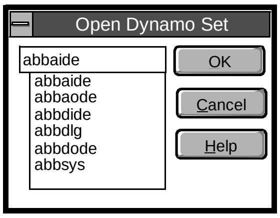


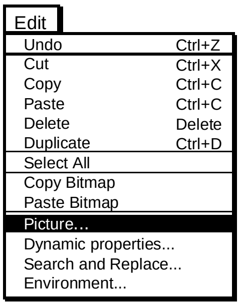
Figure 3-7. AdvaCommand templates in the system display hierarchical model

3.5.6 Reusable graphic objects, Dynamos

ABB has provided you with a set of predefined, reusable, graphic wizards called Dynamos to help you construct operator displays.



A Dynamo can be an object, link, or group of objects with general properties that make them usable in various situations and displays. Using Dynamos saves time and makes drawing your displays easier. It also creates a consistent design, especially if you use the Advant OCS designed Dynamos, throughout your operator displays.



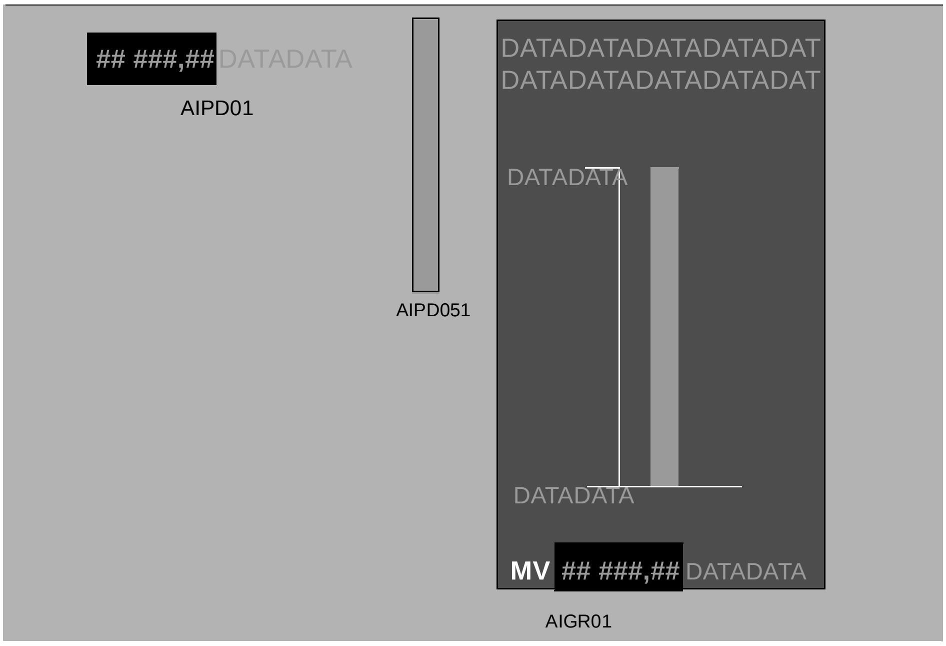
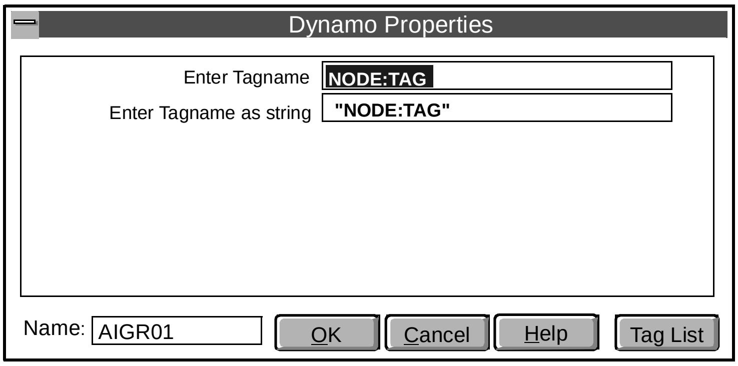


Figure 3-8. Example of a predefined Dynamo set intended for AI objects

Each AdvaSoft Dynamo is ready to use in an Advant OCS, complete with the appropriate dynamic properties. The only additional data you need are the station node number and the tag number which are filled in on the Dynamo Properties dialog (displayed when you paste the Dynamo into your process display), see Figure 3-9.



Enter Tagname NODE:TAG
Enter Tagname as string "NODE:TAG"
Name: AIGR01

Figure 3-9. Dynamo Properties dialog

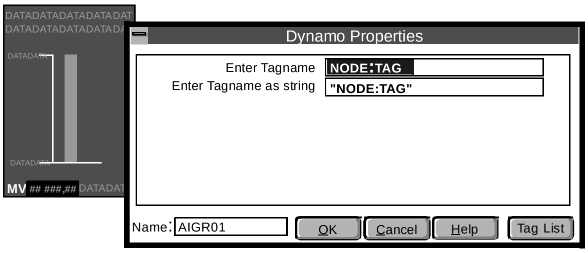
3.5.7 Dynamo structure

Each Dynamo is composed of several different items which are grouped together and then turned into a Dynamo using the command **Create Dynamo**.

Some of the properties are connected to the *Dynamo*, for example, the node and tag properties.

Other properties, like the commands run when selecting the dynamo, are related to the *Group*.

Individual attributes and status bits are related to the individual *Items* in the group.

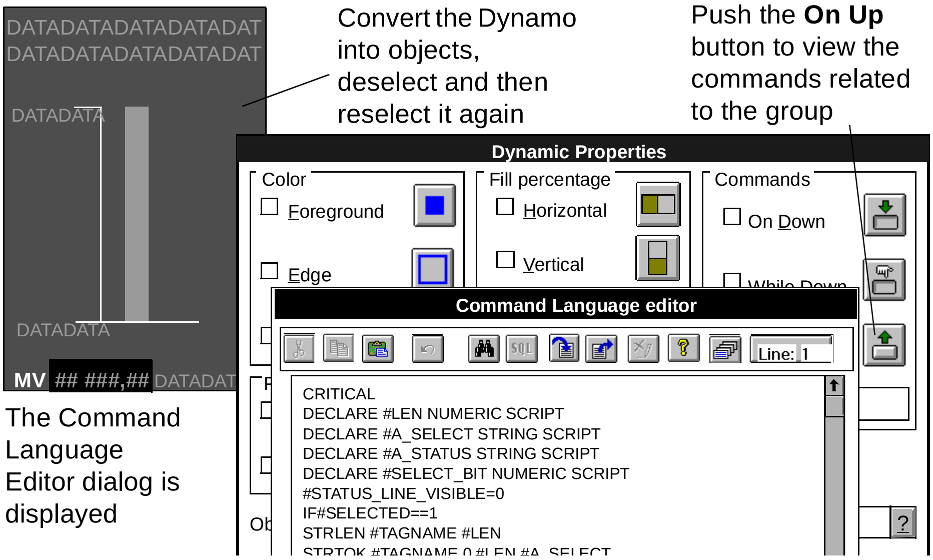


Enter Tagname: NODE:TAG
Enter Tagname as string: "NODE:TAG"
Name: AIGR01

Figure 3-10. Properties related to the dynamo

Each part in the dynamo is given its own dynamic properties to reflect the status for a certain object and a certain object attribute.

To see which attributes each display item in the dynamo is linked to, you can convert, or split, the dynamo into individual items. To do this, select the dynamo and then select the **Convert To Objects** item from the **Dynamos** menu. Double-click on the dynamo after you choose the **Convert to Objects command** and the dialog **Dynamic Properties** is displayed, see Figure 3-11.



Convert the Dynamo into objects, deselect and then reselect it again

Push the **On Up** button to view the commands related to the group

The Command Language Editor dialog is displayed

```
CRITICAL
DECLARE #LEN NUMERIC SCRIPT
DECLARE #A_SELECT STRING SCRIPT
DECLARE #A_STATUS STRING SCRIPT
DECLARE #SELECT_BIT NUMERIC SCRIPT
#STATUS_LINE_VISIBLE=0
IF#SELECTED==1
STRLEN #TAGNAME #LEN
STRTOK #TAGNAME 0 #LEN #A_SELECT
```

Figure 3-11. Command script related to a "converted to objects" dynamo

Select the dynamo which has been converted to objects and then choose the **Ungroup** command from the **Arrange** menu. You can now study each individual item in the dynamo and its relation to the different object attributes.

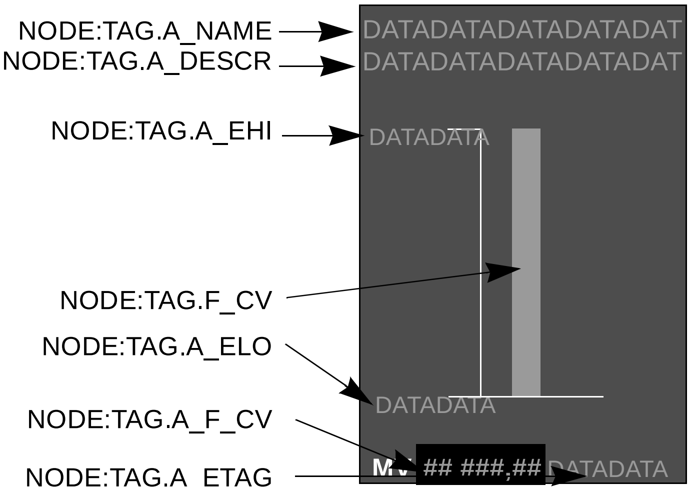
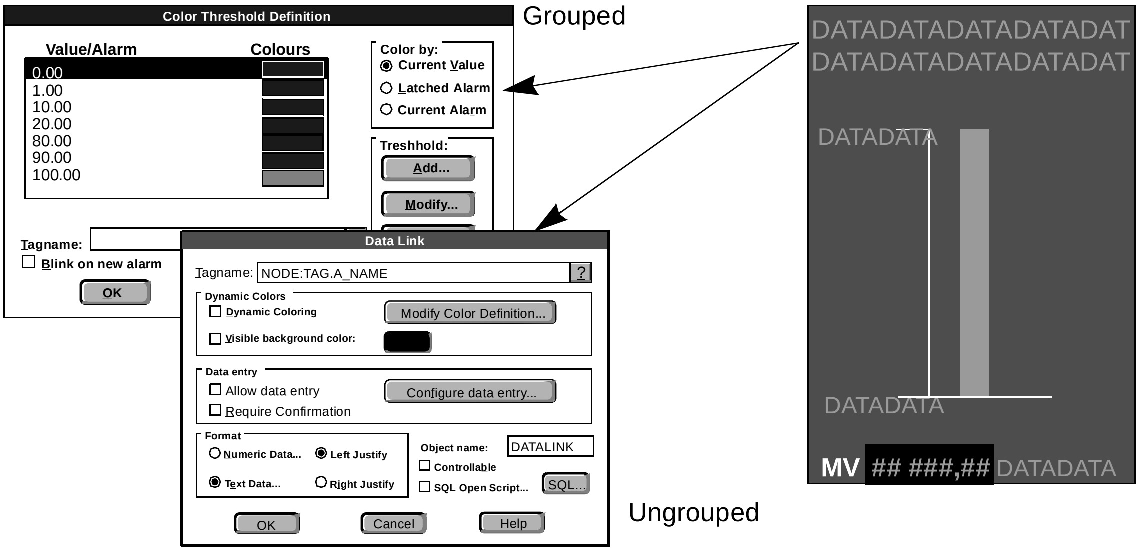


Figure 3-12. Example of a decomposed dynamo

Double-click on each item in the dynamo to display the corresponding configuration dialog. Note that some of the items are doubled and stacked on each other, for example, the name and description rows. Each copy has it own purpose when displaying current object status. Note also that, for example, the name row item is grouped. This is because some of the properties given to the item apply to the group while others apply to the individual item.



Tagname: NODE:TAG.A_NAME

Object name: DATALINK

Figure 3-13. Dialog boxes display depending on state of dynamo item

3.5.7.1 Dynamo related displays

Each AdvaCommand dynamo has a set of displays connected to it which are used for object and object close-up displays. Each display has its own sets of dialog buttons which are used under various conditions. These displays become active when the user selects a object in, for example, a process display.

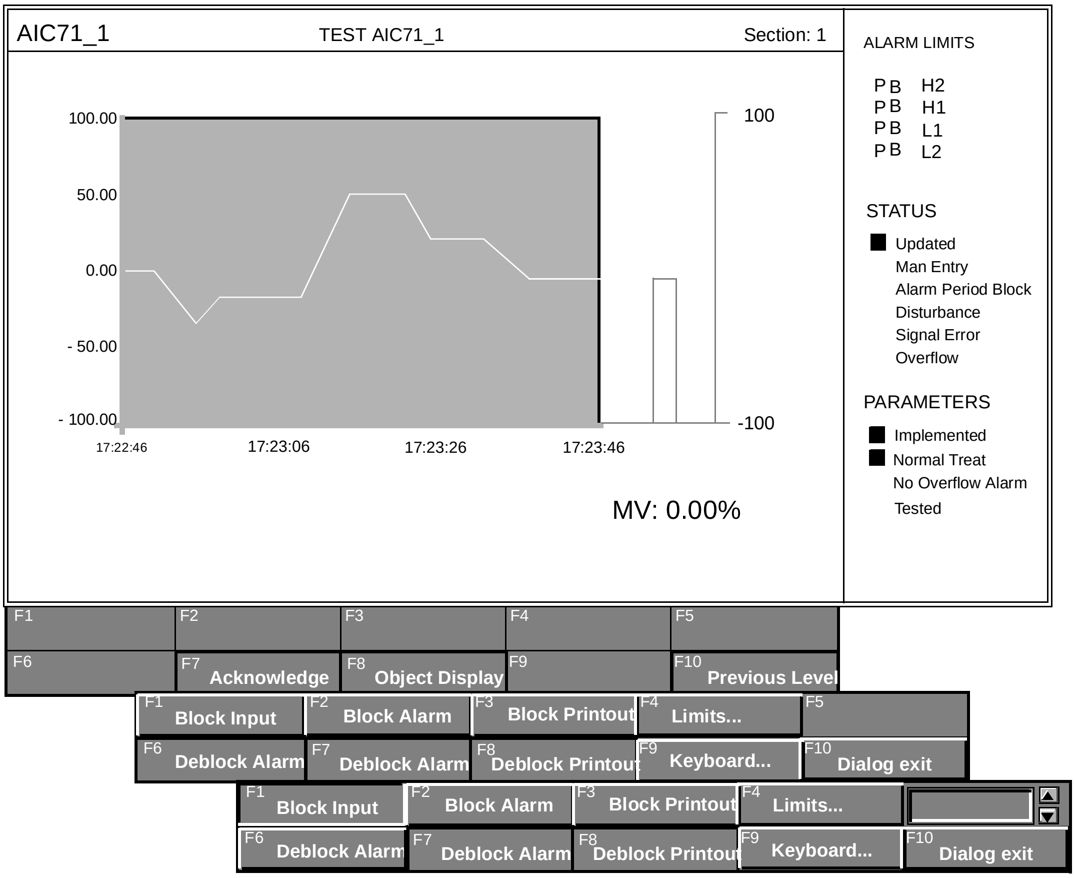


Figure 3-14. Object display with various dialog button sets

3.5.7.2 Display element naming

Names given to the display elements consist of six to eight characters. The first two or three characters indicate the object type.

Table 3-13. Object types

Name	Type
AI	Analog Input
AO	Analog Output
DI	Digital Input
DO	Digital Output
DAT	Data Values

The next two characters indicate the display type at which the display element is aimed.

Table 3-14. Display Type

Name	Type
GR	Group Display
OB	Object Display
WI	Object Close-up display
PD	Process Display

The last two digits are sequential numbers. For example:

AIGR01 is the display element for an analog input signal aimed for a group display.

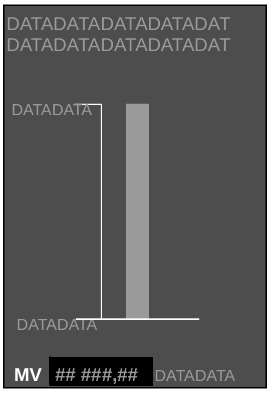


Figure 3-15. Dynamo named AIGR01 intended for a group display

3.5.7.3 Dynamo sets

The tables below list the predefined object display files supplied with AdvaCommand for Windows. When you are designing displays and dialogs, these predefined files offer you uniform, basic appearance and functionality. They are the building blocks, together with the templates and predefined basic displays and dialogs, you need to design your own custom process—and object displays, used together with your application specific library of type circuits in the Advant Controller(s).

Note that you cannot use the predefined dynamos, intended for the Advant OCS process objects, for presentation of data from the FIX database. Some of the Advant OCS object attributes have no corresponding FIX field name.

The Dialog Button Sets are designed to cover the functionality needed for operating process objects via the GCOM interface. If you operate on objects via the Advant Fieldbus 100 interface, some of the available commands will only be handled locally in the AdvaSoft station.

NOTE

The file names listed below are preceded by "_". This distinguishes them from your custom displays. If you save your custom displays without the underscore, they will appear alphabetically in the list after the predefined displays.

The **object displays** include static and dynamic properties for each basic object. Each object has an object display and a close-up display.

Table 3-15. AdvaCommand predefined dynamo sets

Name	Dynamo set content	Description	No. of links
abbaide	AIPD01	Analog Input signal value	8
	AIPD51	Analog Input signal bar graph	7
	AIGR01	Analog Input signal group display	15
abbaode	AOPD01	Analog Output signal value	7
	AOPD51	Analog Output signal bar graph	5
	AOGR01	Analog Output signal group display	11
abbdide	DIPD01	Digital Input signal value (rectangle)	7
	DIGR01	Digital Input signal group display	11
abbdode	DOPD01	Digital Output signal value (rectangle)	5
	DOGR01	Digital Output signal group display	8

Table 3-15. AdvaCommand predefined dynamo sets (Continued)

Name	Dynamo set content	Description	No. of links
abbdlg	DLG BUTTON	Dialog button	—
	ACK BUTTON	Acknowledge button	—
	DIALOG EXIT	Dialog exit button	—
	10 BUTTONS	10 grouped buttons	—
	STOP BUTTON	Stop button	—
	START BUTTON	Start button	—
	START/STOP	Start/Stop buttons	—
abbsys	MENU ITEM	Menu Item	—
	FIRST PAGE	First Page	—
	LAST PAGE	Last Page	—
	PREV PAGE	Previous Picture	—
	NEXT PAGE	Next Page	—
	GROUP SELECT	Group Select (_GRnPGn)	—

You can use a maximum of 800 links in each process display. A standard link to access the value will use two dynamic links for update.

3.5.8 Object attributes reference

The tables below show the object attribute fields available for subscription to AdvaCommand for Windows.

Syntax example: GETVALUE NODE:TAG_NAME.FIELD_NAME

All predefined displays in this chapter contain corresponding field names for presenting dynamic updating.

NOTE

The prefix indicates the data type of the attribute. "A_" indicates an ascii string, and "F_" indicates a floating value. To present a value together with a text string, use "A_". To present the value alone or to write to the remote station, use "F_".

3.5.8.1 Analog Input

Table 3-16. Analog Input attribute fields

Object attribute name	FIX field name	Description	No. of char.	R/W
A_NAME	A_NAME, A_TAG	Object name	20	R
A_DESCRIPTION	A_DESC	Object description	28	R
A_STATUS	–	Status word	–	R
A_UNIT	A_ETAG	Engineering unit	6	R
A_RANGE_MIN	A_ELO	Minimum value of range	11	R
A_RANGE_MAX	A_EHI	Maximum value of range	11	R
A_VALUE	A_CV	Process value	11	R/W
A_ERROR_TREAT		Alarm treatment type of error	11	R
A_NO_OF_DEC		Number of digits right of decimal point	11	R
A_SUBSYSTEM	A_ADI	Process section (Alarm Area)	11	R
A_CLASS		Priority class	11	R
A_HI_LIM2		High alarm limit 2	11	R/W
A_HI_LIM1		High alarm limit	11	R/W
A_LO_LIM1		Low alarm limit	11	R/W
A_LO_LIM2		Low alarm limit 2	11	R/W
A_LIM1_TREAT		Alarm treatment type for limit 1 alarms	11	R
A_LIM2_TREAT	–	Alarm treatment type for limit 2 alarms	11	R
–	A_SELECT A_SEL	Select/deselect	11	R/W
–	A_PROC_UPDATE A_UPDATE	Block/deblock input	11	R/W
–	A_DIST_ALARM A_ALARM	Block/deblock disturbance alarm	11	R/W
–	A_DIST_PRINT A_PRINT	Block/deblock disturbance print	11	R/W
–	A_CUALM	Current alarm status	6	R
–	A_LAALM	Highest severity unacknowledged alarm (latched alarm)	6	R

Table 3-16. Analog Input attribute fields (Continued)

Object attribute name	**FIX field name**	**Description**	**No. of char.**	**R/W**
–	A_NALM	Unacknowledged alarm status. Write to this attribute acknowledges alarms.	5	R/W
F_STATUS	–	Status word (see the different protocols' documentation for an explanation)	3	R
F_RANGE_MIN	F_ELO	Minimum value of range	–	R
F_RANGE_MAX	F_EHI	Maximum value of range	–	R
F_VALUE	F_CV	Process value	–	R/W
F_NO_OF_DEC	–	Number of digits right of decimal point	–	R
F_SUBSYSTEM	F_ADI	Process section (Alarm Area)	–	R
F_CLASS	–	Priority class	–	R
F_HI_LIM2	–	High alarm limit 2	–	R/W
F_HI_LIM1	–	High alarm limit	–	R/W
F_LO_LIM1	–	Low alarm limit	–	R/W
F_LO_LIM2	–	Low alarm limit 2	–	R/W
F_LIM1_TREAT	–	Alarm treatment type for limit 1 alarms	–	R
F_LIM2_TREAT	–	Alarm treatment type for limit 2 alarms	–	R
–	F_SELECT F_SEL	Select/deselect	–	R/W
–	F_PROC_UPDATE F_UPDATE	Block/deblock input	–	R/W
A_DIST_ALARM	F_DIST_ALARM F_ALARM	Block/deblock disturbance alarm	–	R/W
–	F_DIST_PRINT F_PRINT	Block/deblock disturbance print	–	R/W
–	F_NALM	Unacknowledged alarm status. Write to this attribute acknowledges alarms.	–	R/W

3.5.8.2 Analog Output

Table 3-17. Analog Output attribute field names

Object attribute name	FIX field name	Description	No. of char.	R/W
A_NAME	A_NAME	Object name	20	R
A_DESCRIPTION	A_DESC	Object description	28	R
A_STATUS	–	Status word	–	R
A_UNIT	A_ETAG	Engineering unit	6	R
A_RANGE_MIN	A_ELO	Minimum value of range	11	R
A_RANGE_MAX	A_EHI	Maximum value of range	11	R
A_VALUE	A_CV	Process value	11	R/W
A_MIN_LIM	–	The lowest allowed value given as a command	11	R/W
A_MAX_LIM	–	The highest allowed value given as a command	11	R/W
A_START_VALUE	–	The cold start value	11	R
A_ERROR_TREAT	–	Alarm treatment type for errors	11	R
A_NO_OF_DEC	–	Number of digits right of decimal point	11	R
A_SUBSYSTEM	A_ADI	Process section (Alarm Area)	11	R
A_CLASS	–	Priority class	11	R
–	A_SELECT A_SEL	Select/deselect	11	R/W
–	A_DIST_ALARM A_ALARM	Block/deblock disturbance alarm	11	R/W
–	A_DIST_PRINT A_PRINT	Block/deblock disturbance print	11	R/W
–	A_MANUAL A_MAN	Manual/auto mode	11	R/W
	A_OUTP_BLK	Block/Deblock output	11	R/W
F_STATUS	–	Status word (see the different protocols' documentation for an explanation)	–	R
F_RANGE_MIN	F_ELO	Minimum value of range	–	R

Table 3-17. Analog Output attribute field names (Continued)

Object attribute name	FIX field name	Description	No. of char.	R/W
F_RANGE_MAX	F_EHI	Maximum value of range	–	R
F_VALUE	F_CV	Process value	–	R/W
F_MIN_LIM	–	The lowest allowed value given as a command	–	R/W
F_MAX_LIM	–	The highest allowed value given as a command	–	R/W
F_START_VALUE	–	The cold start value	–	R
F_ERROR_TREAT	–	Alarm treatment type for errors	–	R
F_NO_OF_DEC	–	Number of digits right of decimal point	–	R
F_SUBSYSTEM	F_ADI	Process section (Alarm Area)	–	R
F_CLASS	–	Priority class	–	R
–	F_SELECT F_SEL	Select/deselect	–	R/W
–	F_DIST_ALARM F_ALARM	Block/deblock disturbance alarm	–	R/W
–	F_DIST_PRINT F_PRINT	Block/deblock disturbance print	–	R/W
–	F_MANUAL F_MAN	Manual/auto mode	–	R/W

3.5.8.3 Digital Input

Table 3-18. Digital Input attribute field names

Object attribute name	FIX field name	Description	No. of char.	R/W
A_NAME	A_NAME, A_TAG	Object name	20	R
A_DESCRIPTION	A_DESC	Object description	28	R
A_STATUS	–	Status word bit set	–	R/W
A_ERROR_TREAT	–	Alarm treatment type for errors	11	R
A_VALUE_TREAT	–	Alarm treatment type for value alarms	11	R

Table 3-18. Digital Input attribute field names (Continued)

Object attribute name	**FIX field name**	**Description**	**No. of char.**	**R/W**
A_SUBSYSTEM	A_ADI	Process section (Alarm Area)	11	R
A_CLASS	–	Priority class	11	R
–	A_SELECT A_SEL	Select/deselect	11	R/W
–	A_PROC_UPDATE A_UPDATE	Block/deblock input	11	R/W
–	A_DIST_ALARM A_ALARM	Block/deblock disturbance alarm	11	R/W
–	A_DIST_PRINT A_PRINT	Block/deblock disturbance print	11	R/W
–	A_DVAL A_CV	Digital value bit in status word	11	R
–	A_CUALM	Current alarm status	6	R
–	A_LAALM	Highest severity unacknowledged alarm (latched alarm)	6	R
–	A_NALM	Unacknowledged alarm status	5	
F_STATUS	–	Status word bit-set (see the different protocols' documentation for an explanation)	–	R/W
F_ERROR_TREAT	–	Alarm treatment type for errors	–	R
F_VALUE_TREAT	–	Alarm treatment type for value alarms	–	R
F_SUBSYSTEM	F_ADI	Process section (Alarm Area)	–	R
F_CLASS	–	Priority class	–	R
–	F_SELECT F_SEL	Select/deselect	–	R/W
–	F_PROC_UPDATE F_UPDATE	Block/deblock input	–	R/W
–	F_DIST_ALARM F_ALARM	Block/deblock disturbance alarm	–	R/W
–	F_DIST_PRINT F_PRINT	Block/deblock disturbance print	–	R/W
–	F_DVAL F_CV	Digital value bit in status word	–	R

3.5.8.4 Digital Output

Table 3-19. Digital Output attribute field names

Object attribute name	FIX field name	Description	No. of char.	R/W
A_NAME	A_NAME, A_TAG	Object name	20	R
A_DESCRIPTION	A_DESC	Object description	28	R
A_STATUS	–	Status word bit-set	–	R/W
A_ERROR_TREAT	–	Alarm treatment type for errors	11	R
A_SUBSYSTEM	A_ADI	Process section (Alarm Area)	11	R
A_CLASS	–	Priority class	11	R
–	A_SELECT A_SEL	Select/deselect	11	R/W
–	A_DIST_ALARM A_ALARM	Block/deblock disturbance alarm	11	R/W
–	A_DIST_PRINT A_PRINT	Block/deblock disturbance alarm	11	R/W
–	A_MANUAL A_MAN	Auto/manual status	11	R/W
–	A_DVAL, A_CV	Digital value bit in status word	11	R/W
–	A_OUTP_BLK	Block/Deblock output	11	R/W
F_STATUS	–	Status word bit-set (see the different protocols' documentation for an explanation)	–	R/W
F_ERROR_TREAT	–	Alarm treatment type for errors	–	R
F_SUBSYSTEM	F_ADI	Process section (Alarm Area)	–	R
F_CLASS	–	Priority class	–	R
–	F_SELECT F_SEL	Select/deselect	–	R/W
–	F_DIST_ALARM F_ALARM	Block/deblock disturbance alarm	–	R/W
–	F_DIST_PRINT F_PRINT	Block/deblock disturbance alarm	–	R/W

Table 3-19. Digital Output attribute field names (Continued)

Object attribute name	FIX field name	Description	No. of char.	R/W
–	F_MANUAL F_MAN	Auto/manual status	–	R/W
–	F_DVAL, F_CV	Digital value bit in status word	11	R/W

3.5.8.5 DAT

Table 3-20. DAT attribute field names

Object attribute name	Object attribute name	FIX field name	Description
A_NAME	A_NAME	A_NAME A_TAG	Name (string length = 20)
A_VALID	F_VALID	–	Update successful = 1, unsuccessful = 0
A_B0_VAL	F_B0_VAL	–	Bit no. 0 in a Boolean DAT
A_B1_VAL	F_B1_VAL	–	Bit no. 1 in a Boolean DAT
A_B2_VAL	F_B2_VAL	–	Bit no. 2 in a Boolean DAT
A_B3_VAL	F_B3_VAL	–	Bit no. 3 in a Boolean DAT
A_B4_VAL	F_B4_VAL	–	Bit no. 4 in a Boolean DAT
A_B5_VAL	F_B5_VAL	–	Bit no. 5 in a Boolean DAT
A_B6_VAL	F_B6_VAL	–	Bit no. 6 in a Boolean DAT
A_B7_VAL	F_B7_VAL	–	Bit no. 7 in a Boolean DAT
A_B8_VAL	F_B8_VAL	–	Bit no. 8 in a Boolean DAT
A_B9_VAL	F_B9_VAL	–	Bit no. 9 in a Boolean DAT
A_B10_VAL	F_B10_VAL	–	Bit no. 10 in a Boolean DAT
A_B11_VAL	F_B11_VAL	–	Bit no. 11 in a Boolean DAT
A_B12_VAL	F_B12_VAL	–	Bit no. 12 in a Boolean DAT
A_B13_VAL	F_B13_VAL	–	Bit no. 13 in a Boolean DAT
A_B14_VAL	F_B14_VAL	–	Bit no. 14 in a Boolean DAT
A_B15_VAL	F_B15_VAL	–	Bit no. 15 in a Boolean DAT
A_B16_VAL	F_B16_VAL	–	Bit no. 16 in a Boolean DAT

Table 3-20. DAT attribute field names (Continued)

Object attribute name	**Object attribute name**	**FIX field name**	**Description**
A_B17_VAL	F_B17_VAL	–	Bit no. 17 in a Boolean DAT
A_B18_VAL	F_B18_VAL	–	Bit no. 18 in a Boolean DAT
A_B19_VAL	F_B19_VAL	–	Bit no. 19 in a Boolean DAT
A_B20_VAL	F_B20_VAL	–	Bit no. 20 in a Boolean DAT
A_B21_VAL	F_B21_VAL	–	Bit no. 21 in a Boolean DAT
A_B22_VAL	F_B22_VAL	–	Bit no. 22 in a Boolean DAT
A_B23_VAL	F_B23_VAL	–	Bit no. 23 in a Boolean DAT
A_B24_VAL	F_B24_VAL	–	Bit no. 24 in a Boolean DAT
A_B25_VAL	F_B25_VAL	–	Bit no. 25 in a Boolean DAT
A_B26_VAL	F_B26_VAL	–	Bit no. 26 in a Boolean DAT
A_B27_VAL	F_B27_VAL	–	Bit no. 27 in a Boolean DAT
A_B28_VAL	F_B28_VAL	–	Bit no. 28 in a Boolean DAT
A_B29_VAL	F_B29_VAL	–	Bit no. 29 in a Boolean DAT
A_B30_VAL	F_B30_VAL	–	Bit no. 30 in a Boolean DAT
A_B31_VAL	F_B31_VAL	–	Bit no. 31 in a Boolean DAT
A_IW_VAL	F_IW_VAL	–	Integer DAT
A_IL_VAL	F_IL_VAL	–	Integer long DAT
A_R_VAL	–	A_CV	Real DAT
A_VAL_TYPE	F_VAL_TYPE		Type of DAT 1=Boolean, 2=Integer, 3=Integer Long, 4=Real
–	F_R_VAL	F_CV	Real DAT

3.5.9 Object subscription methods

There are a variety methods you can use to subscribe for objects:

- In pictures with default refresh rate (0.1 s), the default subscription cycle time is set to nine seconds. For further information see ***AdvaCommand View User's Guide***.
- In pictures with refresh rate other than 0.1 s, the default subscription cycle time is equal to the picture refresh rate. For further information, see ***AdvaCommand View User's Guide***.
- You can override the above-described subscription cycle time by specifying the cycle time in the tag name, see below.
- For command scripts, the default subscription cycle time is set to nine seconds if no subscription rate is specified in the tagname. For further information see ***AdvaCommand Command Language User's Guide***.
- SQL Command scripts, for further information see ***AdvaCommand Command Language User's Guide***.

To subscribe for objects, you must specify **NODE**, **TAGNAME** and **FIELD**, either directly or using variables. The available fields are listed in Section 3.5.8, Object attributes reference below.

Syntax example using default direct subscription: `NODE1:AI1.A_CUALM`

Syntax example using direct subscription: `NODE1:`**`'AI1;2T'`**`.A_CUALM`
where 2T is the cyclic subscription time (in this case 2*100 msec). The different tag cyclic subscription times are T=0.1 seconds, S=seconds and M=minutes.

Syntax example using variables: `GETVAL #VARIABLE #RESULT`
where #VARIABLE must contain a valid NODE:TAG.FIELD specification, for example, NODE1:AI1.A_CUALM. The result of the GETVAL command is stored in #RESULT, and it is the variable #RESULT which should be printed out on the screen, etc.

3.5.9.1 Specifying subscription update time

There are three ways you can specify the subscription update time for a particular attribute:

- When you are making a data link, you can specify the subscription update time explicitly by entering it as a part of the object name. The syntax is:

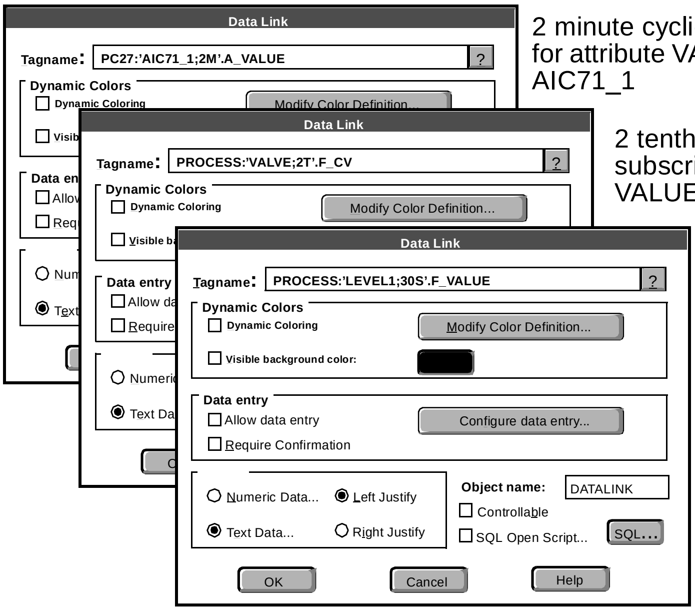
  ```
  NODE:'TAGNAME;type'.FIELD
  ```

  Where NODE specifies a valid node name, TAGNAME specifies a valid object name and FIELD specifies a valid attribute. Note that you must enter the inverted commas and semicolons.

  The syntax for type is:

E	Event subscription
*n*T	Cyclic time in tenth of second; *n* is a number range [1…9]
*nn*S	Cyclic time in seconds; *nn* is a number range [1…59]
*nn*M	Cyclic times in minutes; *nn* is a number range [1…59]
*n*H	Cyclic time in hours; *n* is a number range [1…9]
nn	Cyclic time in seconds; *nn* is a number range [1…59]

See Figure 3-16 for examples of how to fill in the **Data Link** dialog..



Tagname: PC27:'AIC71_1;2M'.A_VALUE

2 minute cyclic subscription for attribute VALUE in object AIC71_1

Tagname: PROCESS:'VALVE;2T'.F_CV

2 tenths of a second cyclic subscription for attribute VALUE in object LEVEL1

Tagname: PROCESS:'LEVEL1;30S'.F_VALUE

Dynamic Colors
☐ Dynamic Coloring
☐ Visible background color: ■

Data entry
☐ Allow data entry
☐ Require Confirmation

○ Numeric Data... ● Left Justify
● Text Data... ○ Right Justify

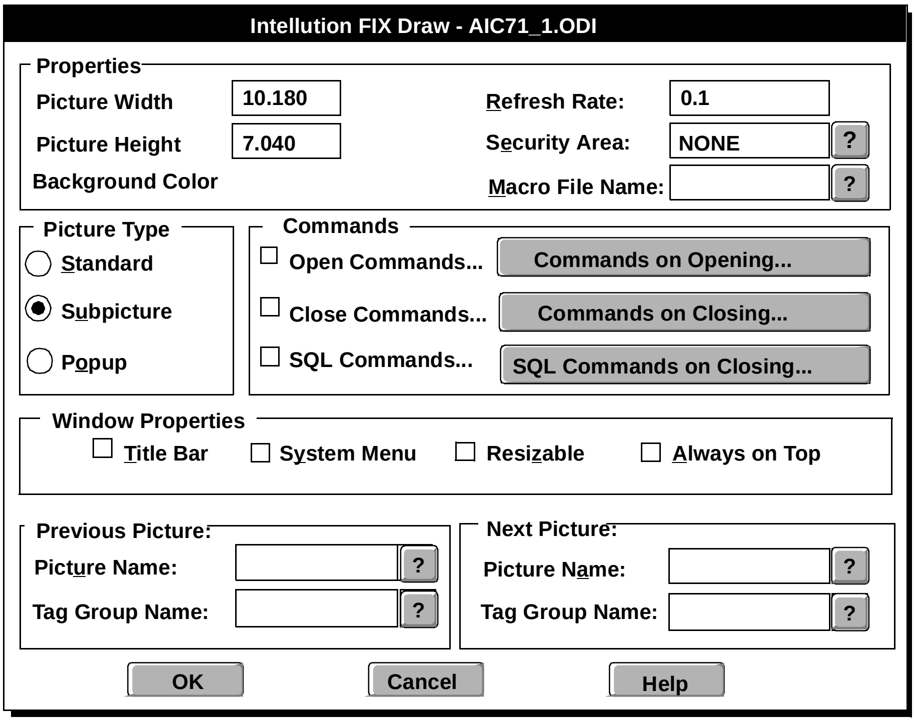
Object name: DATALINK
☐ Controllable
☐ SQL Open Script...

30 second cyclic subscription for attribute VALUE in object LEVEL1

Figure 3-16. Examples of subscription time notations

If you want to use a character other than the semicolon as a separator between the object name and the subscription time, you can alter the **SubSeparator** parameter in the **ADVASOFT.INI** configuration file.

- If you do not enter a specific subscription time in a data link, for example, you specify only NODE, TAGNAME and *x*_FIELD, the time entered as **Refresh Rate** in the **Picture** dialog is used as the subscription time. The picture refresh rate is only used as subscription time if a specific picture refresh rate has not been entered (other than the default, 0.1 seconds). This allows you to set the individual subscription times for the displays.



Intellion FIX Draw - AIC71_1.ODI

Properties
Picture Width: 10.180
Picture Height: 7.040
Background Color
Refresh Rate: 0.1
Security Area: NONE
Macro File Name:

Picture Type
○ Standard
● Subpicture
○ Popup

Commands
☐ Open Commands...
☐ Close Commands...
☐ SQL Commands...

Window Properties
☐ Title Bar ☐ System Menu ☐ Resizable ☐ Always on Top

Previous Picture:
Picture Name:
Tag Group Name:

Next Picture:
Picture Name:
Tag Group Name:

Figure 3-17. The Picture dialog

- If you do not enter any picture refresh rate (it is still 0.1 seconds), the value of the SubScript parameter, found in the ADVASOFT.INI file, is used as the subscription time. The default value of this parameter is 10 seconds.

3.5.9.2 Specifying individual bits within status words

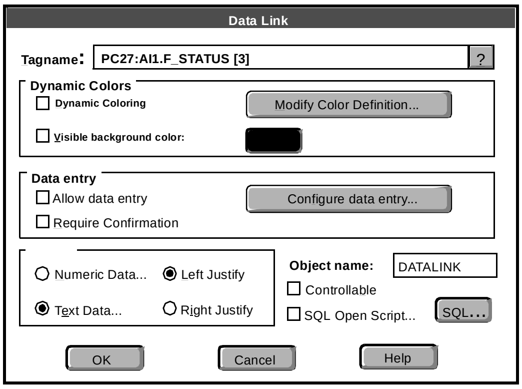
As a system constructor, from time to time you must be able to read certain bits within a status word for deciding the properties of a display object. You may want to change the color for a display object, for example, when the status bit changes value.

To read individual bits, use a special format in the attribute name. The format is:

```
F_status[bit] or A_status[bit]
```

where `attribute` is replaced with the attribute `name and bit` with a number in the range [1…32].

In Figure 3-18 an example of a filled-in dialog is displayed.



Data Link

Tagname: PC27:AI1.F_STATUS [3]

Dynamic Colors
☐ Dynamic Coloring
☐ Visible background color:

Data entry
☐ Allow data entry
☐ Require Confirmation

○ Numeric Data... ● Left Justify
● Text Data... ○ Right Justify

Object name: DATALINK
☐ Controllable
☐ SQL Open Script...

Figure 3-18. Example of a filled-in bit subscription

This example addresses bit 3 in attribute STATUS of an object called AI1.

If you want to use a character other than a straight bracket as the separator between the attribute name and the bit number, you can alter the BitSeparator parameter in the ADVASOFT.INI configuration file.

3.5.9.3 Creating and using custom Dynamos

Refer to the ***Draw*** manual for an introduction to using dynamos. Use the following procedure to create an object and link the object to a close-up display, object display and process dialog. The example below describes how to create an analog value dynamo.

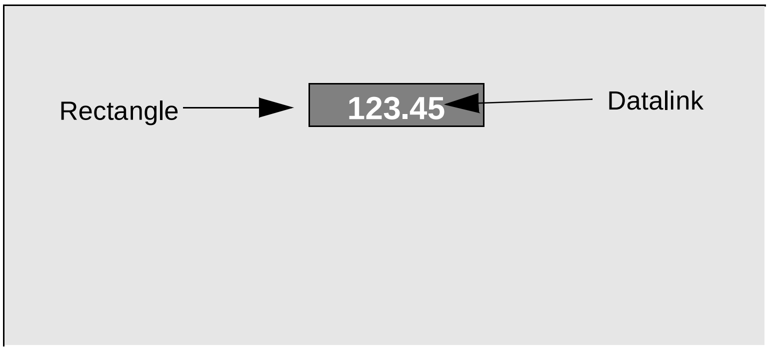
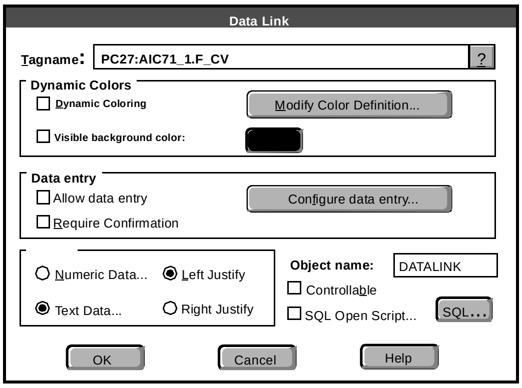


Figure 3-19. Analog value display example

1. Create a data link.



Data Link

Tagname: PC27:AIC71_1.F_CV

Dynamic Colors
☐ Dynamic Coloring
☐ Visible background color:

Data entry
☐ Allow data entry
☐ Require Confirmation

○ Numeric Data... ● Left Justify
● Text Data... ○ Right Justify

Object name: DATALINK
☐ Controllable
☐ SQL Open Script...

Figure 3-20. Data Link display

a. Give the tagname **NODE:TAG.F_CV**.

NOTE

This is a non-existent tagname. When you are asked **Use anyway?**, click **Yes** to confirm. This dialog is displayed when you use non-existent tagnames and is not described further in this example.

b. Click the **Dynamic coloring** or **Modify Color Definition** button (the **Color Threshold Definition** dialog box is loaded).

c. Click the **current alarm** button.

d. Click the **blink on new alarm** button.

e. Enter the generic tagname **NODE:TAG**.

f. Click the **OK** button to exit the dialog.

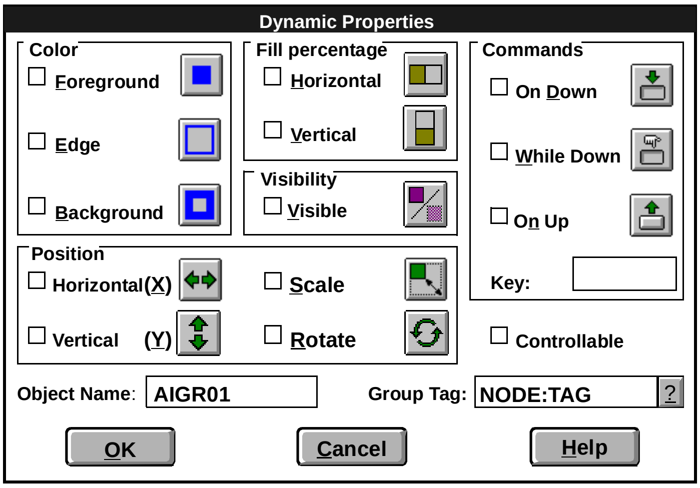
g. Click on the **Numeric Data** button (**Numeric Data** dialog is loaded) in the formatting group.

h. Enter **Whole Digits = 5**
Decimal Digits = 2

i. Click **OK** to exit the dialog.

j. Click **OK** to save changes to the dialog and leave the **Data Link** dialog box.

2. Create a rectangle (see figure). Double-click on the rectangle to load the **Dynamic Properties** dialog box.



Object Name: AIGR01

Group Tag: NODE:TAG

Figure 3-21. Dynamic Properties dialog

   a. Click on the **On Up** button in the **Commands** dialog field to load the Command Language Editor.

   b. In line 1 type: **`#TAGNAME = “NODE:FIELD”`**

   c. In line 2 type: **`REPLACEPIC CLOSEUP _AIWI00`**

   d. Click **OK** to save the command script and exit the editor.

3. Align the rectangle and the datalink in horizontal center (see Figure 3-19).

   a. Group the two objects together.

4. Select the **Dynamo/Create Dynamo** menu command. The **Create Dynamo** dialog box is loaded.

   a. Select an existing dynamo set or create a new set.

   b. Type in a name for your new dynamo, for example, **AIPD02**.

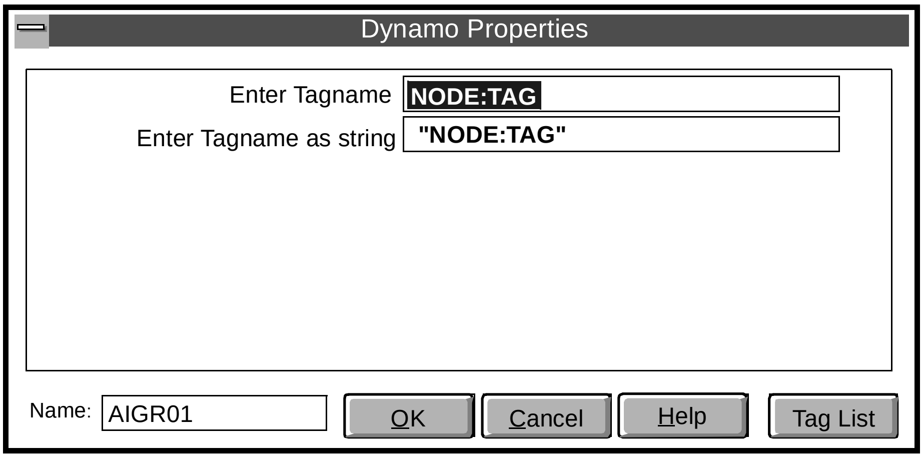
   c. Click **OK** to confirm.

   d. Drop your dynamo into the opened set.

   e. The **User Prompt for the Dynamo** dialog box is loaded.
   In the user prompt field for lines 2, 5 and 6, type the following:
   Line 2 `(“NODE:TAG”):` **`Enter Tagname as string`**
   Line 5 `(NODE:TAG.F_CV):` **`{Enter Tagname}.*`**
   Line 6 `(NODE:TAG.B_CUALM):` **`{Enter Tagname}.*`**

   f. Click **OK** to save changes and leave the **User Prompt for the Dynamo** dialog box.

5. Test your dynamo by dragging it from the dynamo set and dropping it on the process picture. After the dynamo is dropped, a dialog box appears. This dialog should have two fields prompting: **Enter Tagname** and **Enter Tagname as string**.



Enter Tagname: NODE:TAG
Enter Tagname as string: "NODE:TAG"
Name: AIGR01

Figure 3-22. User prompts for a custom made dynamo

   a. In the **Enter Tagname** field, enter node and tag, for example, NODE1:AI1.
   b. In the **Enter Tagname as string** field, enter the **Node** and **Tag** in quotation marks, for example: “NODE1:AI1”.

6. Click **OK** to confirm.

3.5.10 Building custom process displays

When you are building process or object displays, the templates and files listed above include some basic properties such as size, location, colors and basic commands for each display area. These include the display, alarm, dynamic key and small object and logo areas, where relevant.

The ***Command Language*** and ***Draw*** manuals are relevant to this section.

The figure below shows an example of a complete process display.

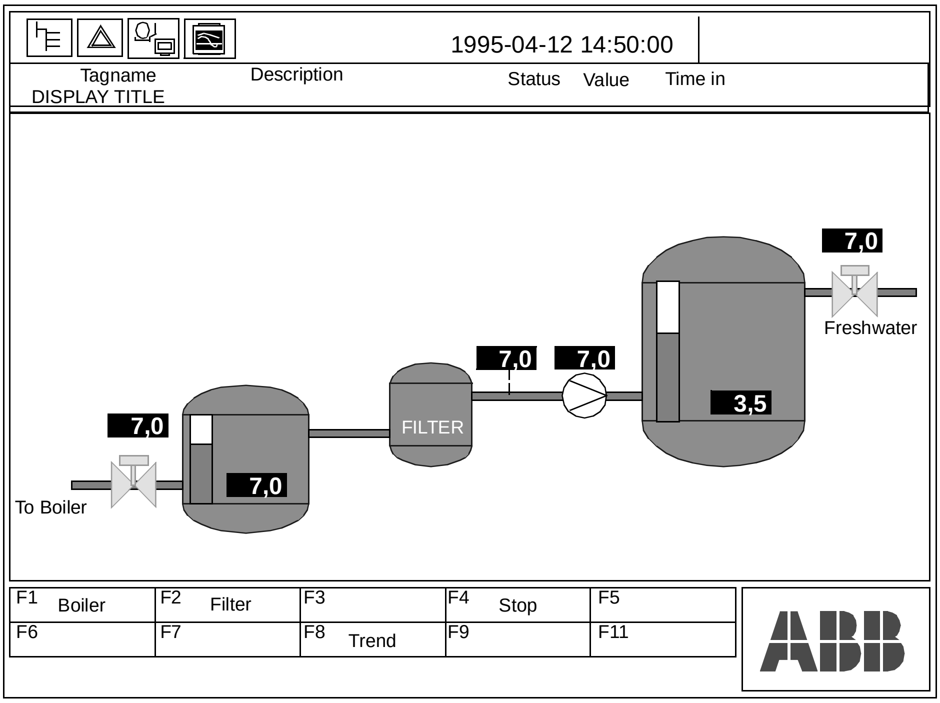


Figure 3-23. Process display example

3.5.10.1 Defining display properties

1. To choose from the template list, choose the **New** item from the Draw **File** menu.

NOTE

The tool box and color box provide shortcuts to frequently used menu items. Open them by choosing the items **Toolbox..**. and **Color box...** from the **Tools** menu.

2. Open the **_PROCDSP** template.
3. Choose **Save as..** from the **File** menu and give the display the name of your choice.

4. To avoid problems with character scaling, etc., when you are creating the process displays, build the displays with the display size for the current resolution. The default resolution is set to the resolution currently used on the PC where AdvaCommand is to be installed. Change the size of the picture by choosing the **Picture...** item from the **Edit** menu and change Picture Width and Picture Height according to Table 3-10.
5. To change properties of the picture, choose the **Picture...** item from the **Edit** menu or the **Picture edit** toolbox button.
6. Choose the **Commands On Opening** button to assign the name you have given the file, and call the display dialog and logo when opening the display, as explained below.
   a. Command line 2: Assigns the “Workscreen” nickname to the current picture.
   Leave unchanged.
   Syntax example: `SETNICKNAME “WORKSCREEN”`
   b. Command line 3: Reference to the picture name.
   Replace the existing text in quotes (include quotes) with a picture name, maximum of eight characters, for the file name reference.
   Syntax example: `#DNAME = “PICTURE NAME MAX 8 CHAR”`
   c. Command line 4: Reference to the picture title.
   Replace the existing text in quotes (include quotes) with a picture title.
   Syntax example: `#DTITLE = “PICTURE TITLE”`
   d. Command line 5: Opens the ABB logo display.
   Replace the name CLOSEUP_LOGO with a display name for a small object closeup display.
   Syntax example: `REPLACEPIC CLOSEUP_LOGO`
   e. Command line 6: Assigns function keys when opening picture.
   Enter name of dialog display you wish to assign to this display.
   Syntax example: `REPLACEPIC BUTTON BUTTON_FILE`
   f. Change the existing, or add more, command lines as required.
   g. Accept by choosing **OK**.
7. Enter **Commands On Closing** if required. As default, there are no commands on closing.
8. Accept changes by choosing **OK**.

3.5.11 Display header configuration

The header display area contains icons with AdvaCommand operator functions.

Add any required custom icons or functions to the **_TOPLINE.odf** picture file from the **Dynamic properties** dialog box.

3.5.11.1 Display title field

The display title is defined as a variable in the MAIN.odf picture file as **#DTITLE**, and placed in the header field under the alarm line. For each picture you define, enter the picture title in the **Commands on opening** definition.

3.5.11.2 Date and time field

1995-04-08 13:58:05

Figure 3-24. Date and time display area

The date and time picture shows the current date and time measured by the system clock. Configure the date format and set the time with the **International and Time** program from the MS-Windows **Control Panel**.

3.5.11.3 Alarm line properties

The predefined alarm line layout and properties are defined in the alarm object, as explained in Section 3.5.17, Alarm summary. The default header alarm line layout is shown below.

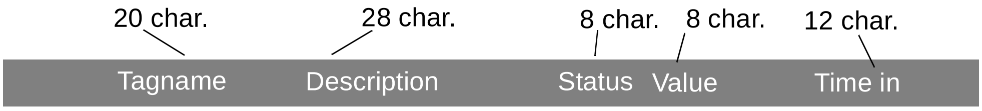


Figure 3-25. Default alarm line layout

To change the properties of this object, open the **_TOPLINE.odf** display file and double-click on the alarm line. Change the layout from the **Column format** dialog box.

If you are using process sections, each object can be assigned a process section. Assign **Alarm area** fields in the **Alarm Summary Configuration** dialog box corresponding to the process section(s) of your process objects.

3.5.12 Defining function keys

3.5.12.1 Dynamic function keys

You can assign each dynamic function key a command or call a new picture with new key functions. Predefined displays have predefined functions which can be changed if necessary.

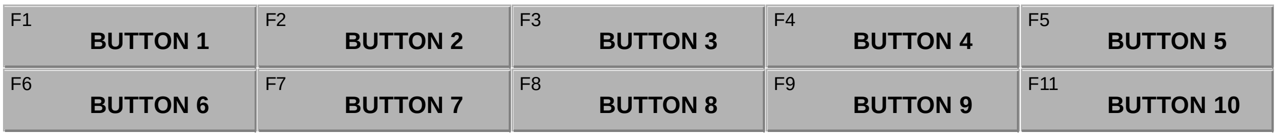


Figure 3-26. Dynamic Function Keys

The static and dynamic properties of the dynamic function keys are predefined in files for each type of display. Assign names to the variables included or design your own functions from scratch. Primarily, there are two types of functions the dynamic function keys are used for: opening new displays and issuing commands to process objects. The **_AOOBDLG.odf** display file contains definitions of display selection, acknowledging alarms and issuing commands to a process object.

The standard sequence for defining dynamic function keys is described below.

1. Choose the **_DISPDLG.odt** template or open the existing dynamic key file, including the dynamic function keys, and save to a new file.
2. If you only want to connect a display to a button, double-click on it and make the necessary changes. If you want to do more, click on the button and choose 'convert to object' from the **Dynamos** menu.
3. Define static properties, for example, texts to the keys with the **Text** item from the **Edit** menu or the toolbox **A** button.
4. Define dynamic properties by double-clicking on a key face to open the **Dynamic Properties** dialog box.
5. Choose the **"!"** button to open the command editor dialog box. Enter the keystroke command(s) required. Some typical examples follow.
   - Acknowledging an alarm: MENU ACKNOWLEDGE_ONE TRUE
   - Issuing a command to a digital object: SETVAL #"VALUE_NAME"
   - Opening a display or dialog: REPLACEPIC "SCREENNAME"
   - Opening an object display for currently selected object: REPLACEPIC WORKSCREEN _AIOBJ (same goes for _AOOBJ, _DIOBJ, _DOOBJ)

3.5.12.2 Keyboard function keys

For each picture with function keys, a ***.KMX** file exists, defined by the **Macro Editor**. This file is loaded to all open pictures every time a dialog (function key picture) opens. It is very important that you load this KMX file after the pictures are opened. Otherwise, the KMX file is only connected to the closed picture and not to the new one. The KMX file consists of only the available function keys. Other commands are taken from the VIEW.KMX.

NOTE

You cannot use **Function key F10** because it is used by Windows.

3.5.13 Defining object close-up properties

The static layout and dynamic fields for the object close-up AI are displayed below.
For reference to all available field names for all object types, see Section 3.5.8, Object attributes reference.

When you open an object close-up, all object information for the object is first refreshed when the variable close-up display, **_AIWI00.ODF**, is opened through the "commands on opening" script. When all values are updated, **_AIWI00.ODF** replaces itself with the object close-up **AIWI01.ODF** for the selected object type.

Figure 3-27 shows the layout of an analog input object close-up display and the sequence in which the displays are opened. The #variables on the figure are globally defined in the **_MAIN.ODF** file.

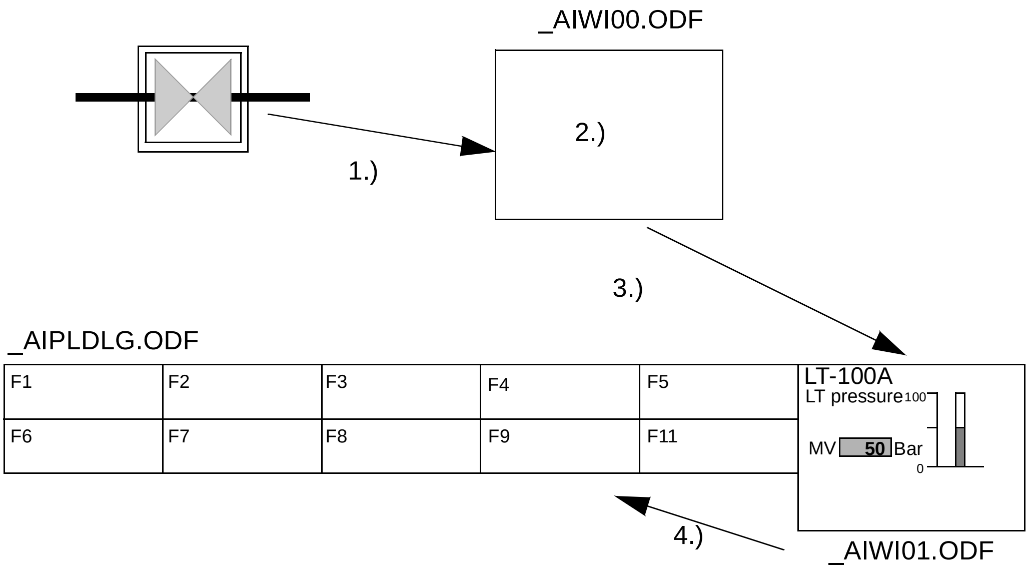


Figure 3-27. Execution sequence for loading process dialog and close-up display

1. The variable close-up opens when you select an object.

2. The variable close-up updates current values to variables and
3. Opens the corresponding dialog and object close-up display.
4. **_AIWI01.ODF** opens the corresponding dialog.

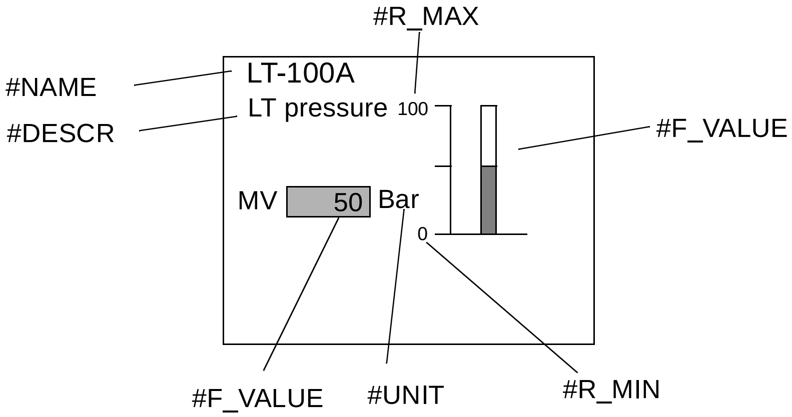


Figure 3-28. Object close-up of an Analog input

The AI variable close-up display executes the following command script upon opening. The **NODE:TAG** field for the selected object is passed to the variable close-up display, to the global variable **#TAGNAME**.

Line 1: SETNICKNAME “CLOSEUP”

This line assigns the nickname CLOSEUP to the window.

Line 2: DECLARE #LEN NUMERIC SCRIPT

This line declares a numeric variable with name #LEN.

Line 3: STRLEN #TAGNAME #LEN

This line reads the number of characters in #TAGNAME and stores it in #LEN. Node and tagname of the last selected object are stored in #TAGNAME. Example: NODE1:LT100.

Line 4: STRTOK #TAGNAME 0 #LEN #BL

This line copies the contents of #TAGNAME into the variable #BL and copies #LEN (#LEN contains the number 11) characters from #TAGNAME start in position 0.
#BL is now = NODE1:LT100.

Line 5: STRCAT #BL ".B_LAALM"

This line concatenates the fieldname “.BL” to #B_LAALM. #B_LAALM is now = NODE1:TT100.B_LAALM. This is a complete NTF for reading the tagname of the selected object.

Line 6: STRTOK #TAGNAME 0 #LEN #F_USE_HI_LIM1

This line copies the contents of #TAGNAME into the variable #F_USE_HI_LIM1 and copies #LEN (#LEN contains the number 11) characters from #TAGNAME start in position 0.
#F_USE_HI_LIM1 is now = NODE1:LT100.

Line 7: STRCAT #F_USE_HI_LIM1 ".F_USE_HI_LIM1"

This line links the fieldname ".F_USE_HI_LIM1" to #F_USE_HI_LIM1. #F_USE_HI_LIM1 is now = NODE1:TT100.A_TAG. This is a complete NTF for reading the tagname of the selected object.

Line 8: GETVAL #F_USE_HI_LIM1 #USE_HI_LIM1

Uses the **GETVAL** command to read the tagname from #USE_HI_LIM1. The tagname is copied into the variable #USE_HI_LIM1. The #USE_HI_LIM1 variable is printed out on the screen using a datalink. The contents of #USE_HI_LIM1 is LT100.

Line 9: STRTOK #TAGNAME 0 #LEN #F_USE_HI_LIM2

Line 10: STRCAT #F_USE_HI_LIM2 ".F_USE_HI_LIM2"

Line 11:GETVAL #F_USE_HI_LIM2 #USE_HI_LIM2

This line reads the description for selected tag. See description of lines 4 to 6.

Line 12:STRTOK #TAGNAME 0 #LEN #F_USE_LO_LIM1

Line 13: STRCAT #F_USE_LO_LIM1 ".F_USE_LO_LIM1"

Line 14: GETVAL #F_USE_LO_LIM1 #USE_LO_LIM1

This line reads engineering unit for selected tag. See description of lines 4 to 6.

Line 15: STRTOK #TAGNAME 0 #LEN #F_USE_LO_LIM2

Line 16: STRCAT #F_USE_LO_LIM2 ".F_USE_LO_LIM2"

Line 17: GETVAL #F_USE_LO_LIM2 #USE_LO_LIM2

This line reads the value of high range for selected tag. See description of lines 4 to 6.

Line 18: STRTOK #TAGNAME 0 #LEN #A_SUB

Line 19: STRCAT #A_SUB ".A_SUBSYSTEM"

Line 20: GETVAL #A_SUB #SUBSYST

This line reads the value of low range for selected tag. See description of lines 4 to 6.

Line 21: STRTOK #TAGNAME 0 #LEN #A_CLASS

Line 22: STRCAT #A_CLASS ".A_CLASS"

This line reads dynamic values from process computer. "GETVAL" is not needed for the F_CV field. See description of lines 4 to 6.

Line 23: GETVAL #A_CLASS #CLASS

Line 24: STRTOK #DNAME 0 8 #OBJECT_DISPLAY

Line 25:STRCAT #AI_STATUS_BITS[0][0] ".F_STATUS[1]"

This line reads the value of subsystem for selected tag. See description of lines 4 to 6.

Line 26: STRCAT #AI_STATUS_BITS[1][0] ".F_STATUS[2]"

Line 27: STRCAT #AI_STATUS_BITS[2][0] ".F_STATUS[3]"

Line 28: STRCAT #AI_STATUS_BITS[3][0] ".F_STATUS[4]"

Line 29: STRCAT #AI_STATUS_BITS[4][0] ".F_STATUS[5]"

Line 30: STRCAT #AI_STATUS_BITS[6][0] ".F_STATUS[7]"

Line 31: STRCAT #AI_STATUS_BITS[8][0] ".F_STATUS[9]"

Line 32: STRCAT #AI_STATUS_BITS[9][0] ".F_STATUS[10]"

Line 33: STRCAT #AI_STATUS_BITS[10][0] ".F_STATUS[11]"

Line 34: STRCAT #AI_STATUS_BITS[11][0] ".F_STATUS[12]"

Line 35: STRCAT #AI_STATUS_BITS[14][0] ".F_STATUS[15]"

Line 36: STRCAT #AI_STATUS_BITS[15][0] ".F_STATUS[16]"

Line 37: STRCAT #AI_STATUS_BITS[16][0] ".F_STATUS[17]"

Line 38: STRCAT #AI_STATUS_BITS[17][0] ".F_STATUS[18]"

Line 39: STRCAT #AI_STATUS_BITS[18][0] ".F_STATUS[19]"

Line 40: STRCAT #AI_STATUS_BITS[22][0] ".F_STATUS[23]"

Line 41: STRCAT #AI_STATUS_BITS[23][0] ".F_STATUS[24]"

This line reads the individual bits in the status attribute.

Line 42: REPLACEPIC BUTTON _AIDLG02

This line replaces the variable close-up with the real close-up for the selected signal.

3.5.13.1 Create a new Close-up window

1. Open an existing close-up display file.
2. Modify or assign dynamic properties and data links to objects according to the following procedure.
   a. Refer to all dynamic properties and/or data links with variables.
   b. Two variables are needed for all fields except the "F_CV" field. Declare your new variables as global scope variables in the "commands on opening" script in the _MAIN.ODF file.
   c. Build up your new reference variable as described in Section 3.5.13, Defining object close-up properties.
      Example:
      Line x. STRTOK #TAGNAME 0 #LEN #NEW_VAR_TAGNAME
      Line y. STRCAT #NEW_VAR_TAGNAME ".*fieldname*"
      Line z. GETVAL #NEW_VAR_TAGNAME #NEW_VAR_VALUE
      Note: if *fieldname* is an F_CV field line, z is not needed. The #NEW_VAR_TAGNAME variable can be used directly in data links.
   d. Replace the last two lines in the script, the REPLACEPIC CLOSEUP "close-up file name", with the right dialog and close-up display names.
3. Save your variable close-up by giving it a new name with the **"Save As..."** menu item.
4. Open an existing object close-up display file.

5. Modify or assign dynamic properties and data links to objects using the following procedure:
   - For all dynamic references, use #variables assigned in the appropriate variable close-up.
6. Save your object close-up by giving it a new name with the **"Save As..."** menu item.

3.5.14 Defining Display menus

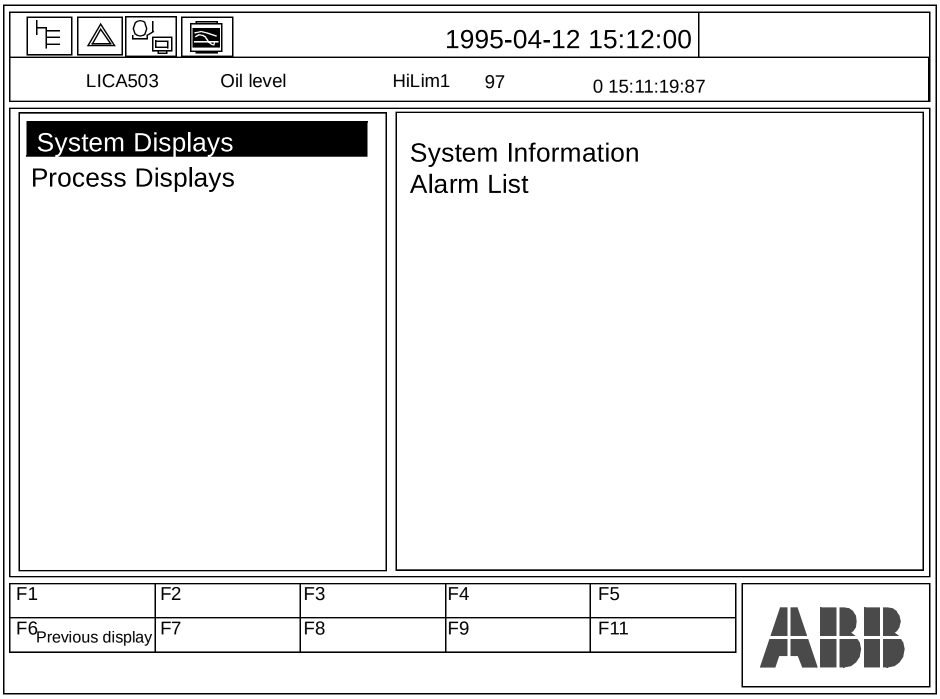


Figure 3-29. Display menu display

3.5.14.1 Predefined Display menu

The predefined **Display** menu consists of two display parts, the **display group** and **display menu list**. If you wish to group the displays, or the number of displays exceeds the display list area, add display lists to the groups to the left.

You can find a predefined display list as shown above in the **_DSPMENU.odf** display file. This opens the corresponding dialog, logo and the **_GR1PG1.odf** display file. After you click the **Process Displays** text, the **_GR2PG1.odf** display file is opened.

3.5.14.2 Adding new groups

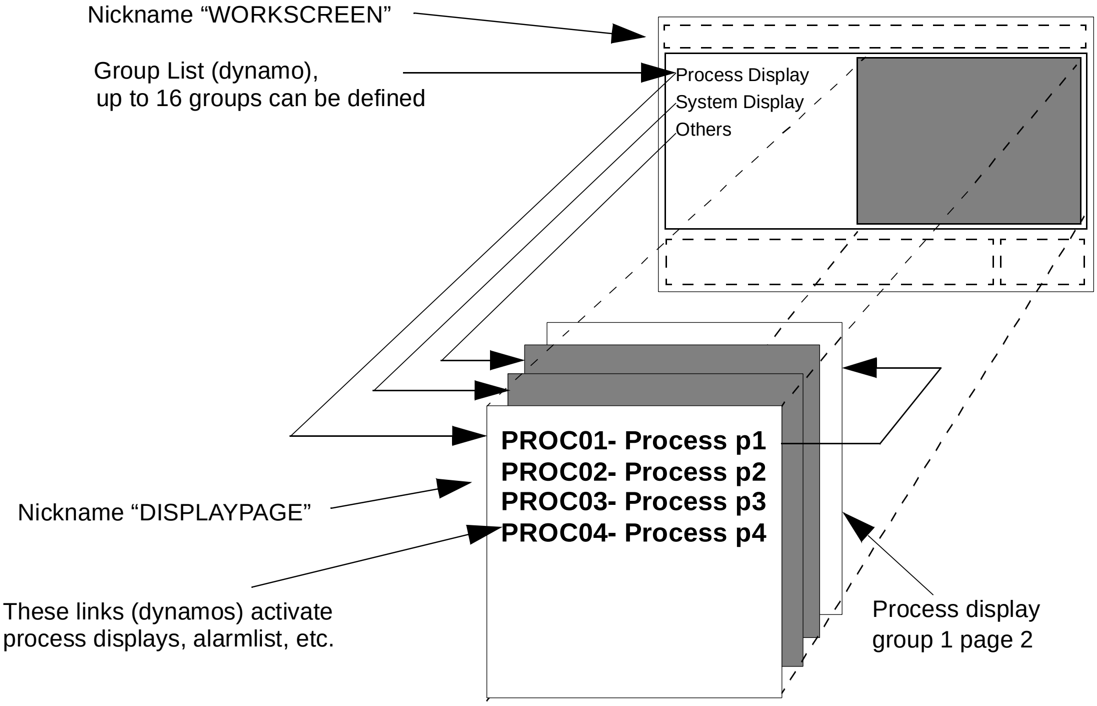


Figure 3-30. Functionality of display menu and group page displays

1. Open the **_DSPMENU.ODF** display file.
2. Double-click on the **group list** area on the left side of the screen to open the **Dynamo Properties** dialog box for the group list.
3. Locate the group(s) you want to add or change. Change the title. Press the **OK** button. The list is made as a dynamo—only the group title can be changed.
4. Change properties of the _DSPMENU picture by choosing the **Picture...** item from the **Edit** menu or the **Picture edit** toolbox button.
5. Choose the **Commands On Opening** button to change the variable **#ITEMS_IN_GR_LIST** to reflect the actual number of items in the group list. You can define a maximum of 16 groups. For example, if you want three items in the group list assigned **#ITEMS_IN_GR_LIST** = 3, the remaining 13 items in the list are invisible at runtime.

NOTE

If the picture is too big to save in Edit Picture..., enable the title bar checkbox.

6. Every item in the list loads a picture group display with a predefined name.
Group 1 or Item 1 loads the display **_GR1PG1.odf** while Item 3 loads **_GR3PG1.odf**.
The command executed when the user clicks on a menu item in runtime is “REPLACEPIC DISPLAYPAGE _GRNPGN.odf”.

3.5.14.3 Creating a new group page

The grouping of display menus into, for example, process sections makes selection quick and eliminates the need to scroll through long lists. The group link loads the first page and paging keys enable you to scroll directly to the next, previous, last and first pages in the group.

You can use the template _GRnPDn to create new pages.

1. Change the **#LASTGROUP** variable in **On Opening commands** in the page display. Choose the **Picture** item from the **Edit** menu (or directly from the toolbox) and open the script editor by clicking the **Commands on Opening** button. Change the **#LASTGROUP** variable to contain the name of the display.
   Example: **#LASTGROUP = “_GR1PG1”**.
2. Click **OK** to save the changes.
3. Change display links by double-clicking on the predefined display link **Name - Description**. Give the name of the process display you want to go to while clicking this link. Fill in the name and description in the **Name and Description** field. Press **OK** button to save changes.
4. Repeat the above steps for all the display links you have on the page.
5. Delete links not in use.
6. Save the display.

A **Name - Description** link is found in the dynamo set **ABBSYS**. You can use this dynamo to add display links to a page display.

NOTE

Page displays called up by display links in the group menu require predefined names. The first page of group 1 must have the name _GR1PG1.odf and the first page of group 16 must have the name _GR16PG1.odf. If you need more than one page for a group, it is easy to keep track of the displays by adding a new page number. For example, in this case, page 2 in group 16 would be called _GR16PG2.odf.

3.5.14.4 Adding paging keys

When a menu list contains more than 16 items or entries, you can add paging keys for scrolling through the pages required to display the list.

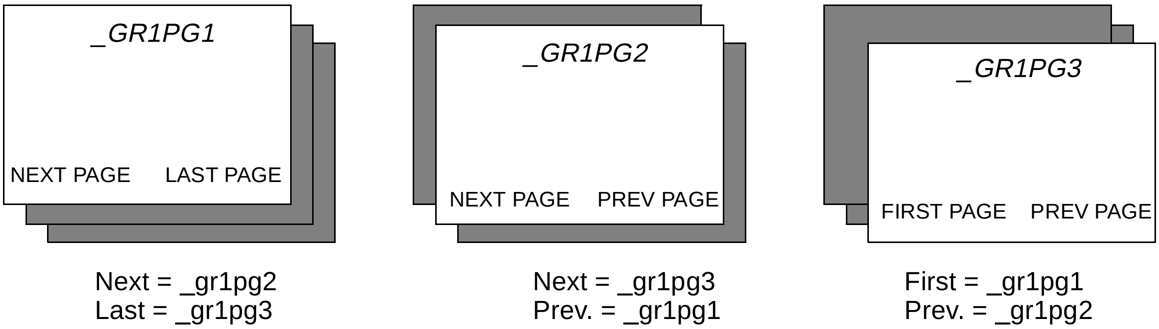


Figure 3-31. Use of paging keys in group page displays

1. Open/create the **_GRnPGn** file you want.
2. Open the dynamo set **ABBSYS** by selecting the Open Set... item from the Dynamo menu.
3. Select the **ABBSYS** dynamo from the list and click the **OK** button to open.
4. Drag the selected dynamo to the desired page display. Lift your finger from the mouse button to drop the dynamo in place. Write the name of the next picture and change button test if required.
5. Click **OK** to confirm.
6. Repeat steps for keys as required.

3.5.15 Group displays

The object group displays have a predefined layout and dynamic fields, as shown below, prepared for groups of 10 object close-up displays.

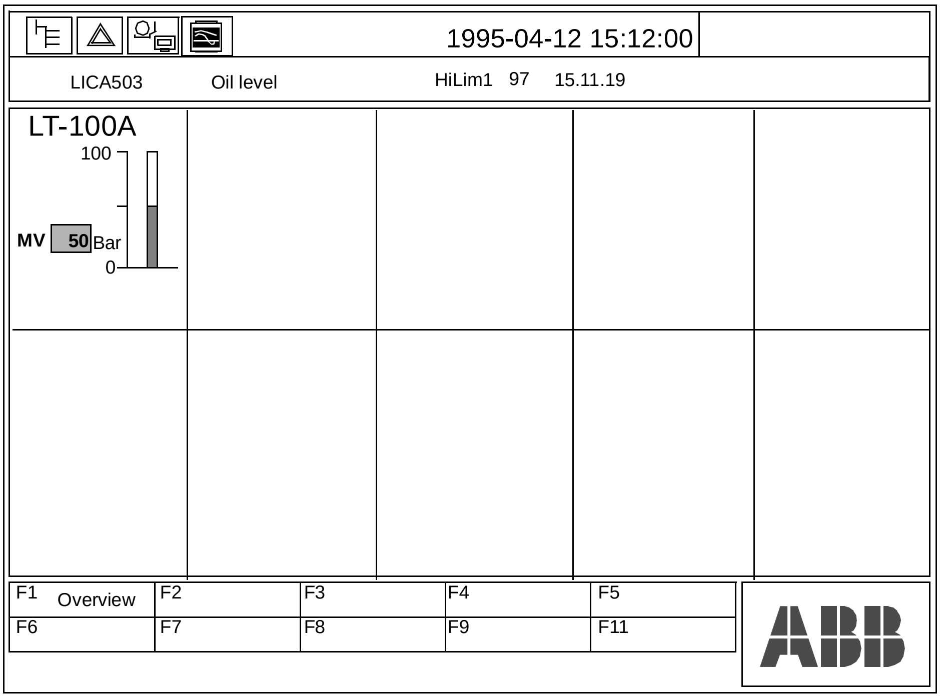


Figure 3-32. Group display layout with examples

3.5.15.1 Designing group displays

1. Select **New** from the **File** menu.
2. Select the **_GRPDSP** template and click the **OK** button to open.
3. Select **Picture** from the **Edit** menu.
4. Choose **Commands on Opening** for a process display.
5. Open the corresponding Dynamo Set, depending on object type, for example, ABBAIDE.
6. Copy necessary group displays on the display as required. Fill in necessary Dynamic properties for the selected dynamo.
7. Align the display element to the grid on the group display.
8. Repeat steps 5 and 6 for all the Group Displays you want on the page for a maximum of 10 objects.
9. Save to new files as required for your application.

3.5.16 Object displays

The predefined object displays include static and dynamic properties for the basic objects AI, AO, DI and DO. The procedure for designing your own type circuit object displays is shown below.

Variables used in the object display are global variables. These variables are assigned when the close-up display is opened. Only those variables not assigned in the definition close-up display for a selected object need to be assigned in the opening script for the object display.

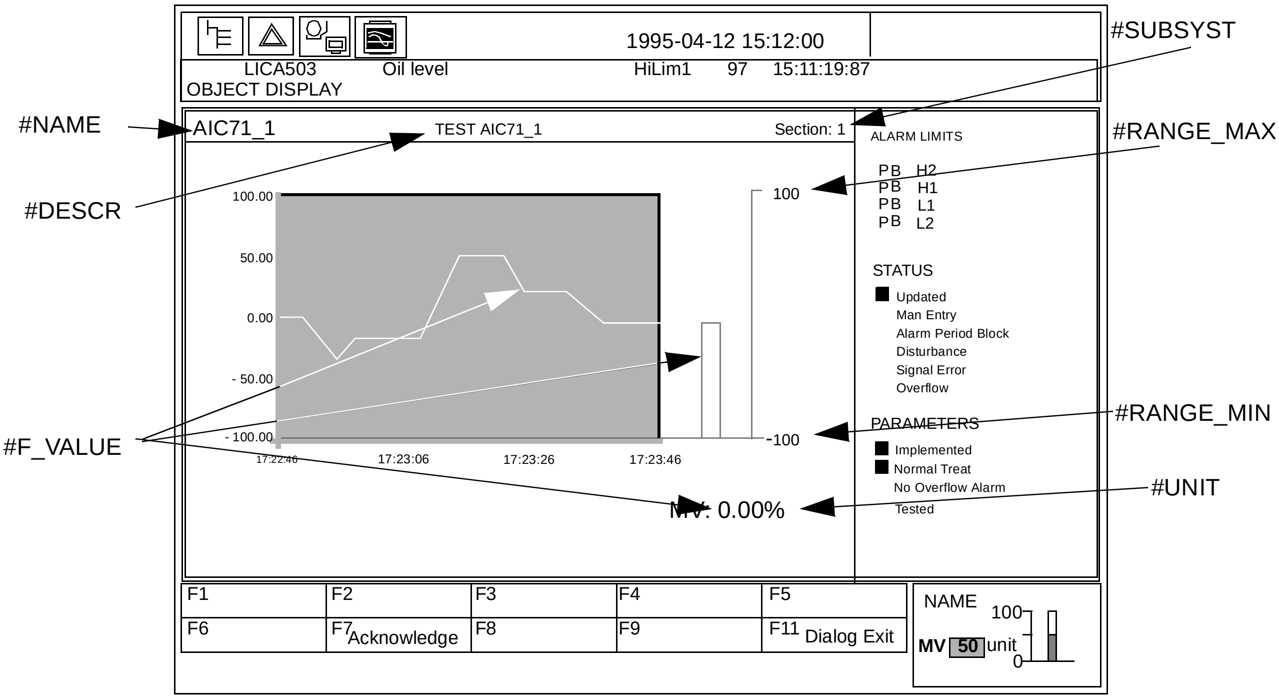


Figure 3-33. Analog object display layout

3.5.16.1 Creating a custom object display

1. Open an existing object display.
2. Modify or assign dynamic properties and data links to objects using the following procedure:
   a. Refer to all dynamic properties and/or data links with variables. If variables are not defined and assigned in the objects close-up window, define them globally in **_MAIN.ODF**.
   b. Build up your new reference variable as described in Section 3.5.13, Defining object close-up properties.
   Example:
   Line x. STRTOK #TAGNAME 0 #LEN #NEW_VAR_TAGNAME
   Line y. STRCAT #NEW_VAR_TAGNAME ".fieldname"
   Line z. GETVAL #NEW_VAR_TAGNAME #NEW_VAR_VALUE

NOTE

If ".fieldname" is an "F_CV" field line, it is not necessary to precede it with the "#" symbol. You can use the #NEW_VAR_TAGNAME variable directly in data links.

3. Save your object display under a unique name.

3.5.16.2 Assigning custom object displays to objects

1. Open the object display for the corresponding object and save to a new file.
2. Enter the object name in the **Commands on opening...** dialog box, then modify the data in the command line editor for the display.

Line 1: SETNICKNAME "WORKSCREEN"

Line 2: STRTOK #DNAME 0 8 #LASTDISPLAY

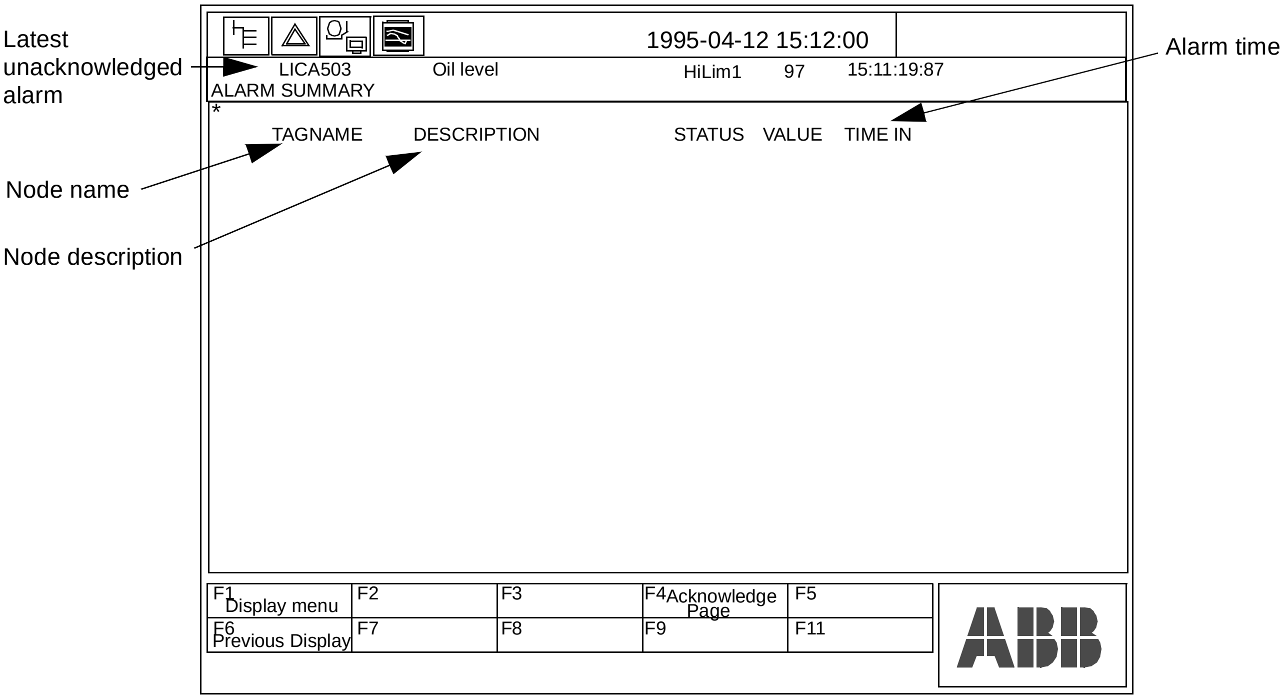
Line 3: #DNAME = "_AIOBJ"

Line 4: #DTITLE = "Object display for AI"

Line 5: REPLACEPIC BUTTON _AIOBDLG

3.5.17 Alarm summary

The supplied predefined alarm summary display and dialog, **_ALLIST.odf** and **_ASUMDLG.odf**, have a predefined layout as shown below.



1995-04-12 15:12:00

LICA503 Oil level HiLim1 97 15:11:19:87

ALARM SUMMARY

Figure 3-34. Alarm summary display

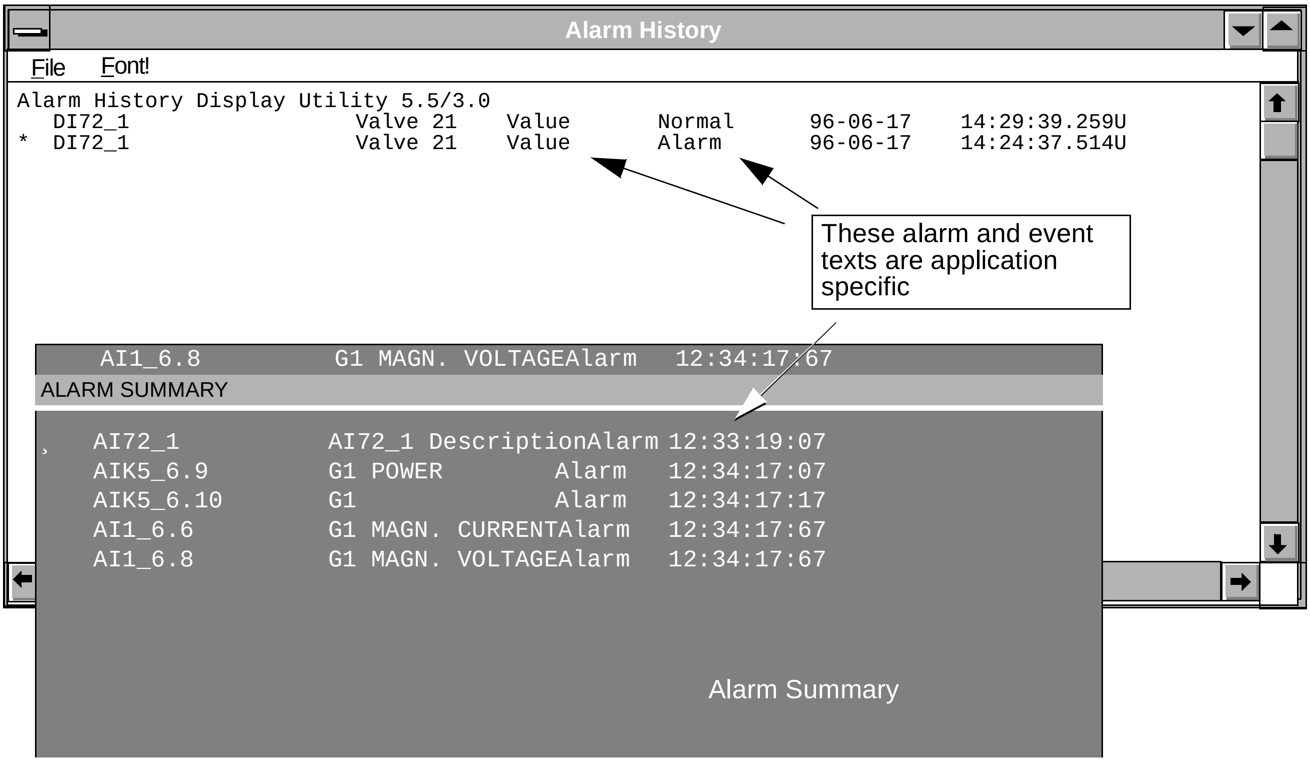
To create your own customized Alarm Summary, see Section 3.5.18, Creating a custom Alarm Summary.

3.5.18 Creating a custom Alarm Summary

You may want to have separate lists for various process sections, to have separate lists for process and system messages, or to define your own custom alarm and event texts.

3.5.18.1 Custom defined Alarm and event treatment texts

As application builder, you can define your own custom alarm and event texts to make them more application specific. These texts are to be displayed in the Alarm Summary and History displays, together with the predefined texts, see Figure 3-35.



```
Alarm History Display Utility 5.5/3.0
   DI72_1                   Valve 21    Value       Normal      96-06-17    14:29:39.259U
*  DI72_1                   Valve 21    Value       Alarm       96-06-17    14:24:37.514U
```

These alarm and event texts are application specific

```
     AI1_6.8          G1 MAGN. VOLTAGEAlarm   12:34:17:67
```

```
    AI72_1           AI72_1 DescriptionAlarm 12:33:19:07
    AIK5_6.9         G1 POWER        Alarm   12:34:17:07
    AIK5_6.10        G1              Alarm   12:34:17:17
    AI1_6.6          G1 MAGN. CURRENTAlarm   12:34:17:67
    AI1_6.8          G1 MAGN. VOLTAGEAlarm   12:34:17:67
```

Figure 3-35. Example of an alarm printout using custom alarm texts

An **Event Message Index** on alarms and events is received from the User API. This index is handled by the Current Alarm Handler. Indexes below 20 are standard indexes that have a predefined meaning. Indexes above 19 are application specific event message indexes that can be configured. See Figure 3-38 for an example of how the Event Message Index is used.

NOTE

To handle application specific event indexes, the databases located in the Controllers and Operator Stations must be configured to handle Event treatment Indexes greater than 19. See ***AdvaInform, Object Types Reference Manual***.

For the *standard* indexes, the Event and Property texts are read from a fixed table in memory.

For the *application specific* Event Treatment Indexes, three configuration files are used, ACEVENT.TXT, ACEVTEXT.TX and ACPRTEXT.TXT. ACEVENT.TXT holds all the alarm text handling while the alarm texts are located in the other two configuration files, see Figure 3-36.

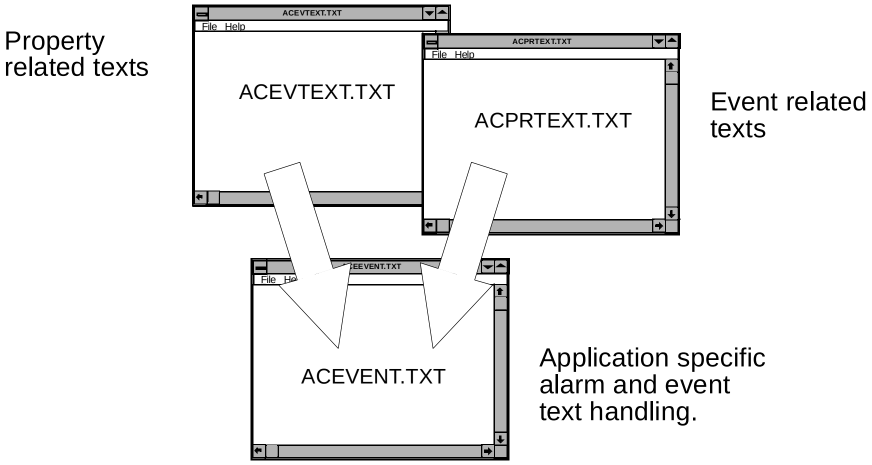


Figure 3-36. Configuration files used for application specific alarm and event text handling

NOTE

You must manually create the ACEVENT.TXT, ACEVTEXT.TXT and ACPRTEXT.TXT files and place them in the \ADVASOFT directory.

3.5.18.2 Parameters used for Alarm and Text handling

The parameters used for alarm text handling are located in the ACEVENT.TXT file, see Figure 3-38. Five parameters control the alarm text display.

- **Event Treatment Word**: This is a four-bit word that configures how the Alarm Handler should treat the event when it occurs and when it normalizes--as an alarm, as an event or to be ignored. The following figure shows how the bits in the four-bit word control the event treatment.

Bit	Description
0	0=Disable treatment, 1=Enable treatment
1	Occurrence treatment; 0=Event, 1=Alarm
2	Normalization treatment; 0=Nothing, 1=Event
3	Event will be sent to Alarm Summary only

Figure 3-37. Explanation of the four-bit Event Treatment Word

You enter the Event Treatment Word in decimal format into the configuration file. The number is then translated into binary format. For example, if you want to set the two first bits to one and the third bit to zero — Enable treatment, Alarm, Nothing (not using the fourth bit) —then you enter the figure 7 (111 in binary format) for the Event Treatment Word.

NOTE

The Event Treatment Word overrides the event reason.

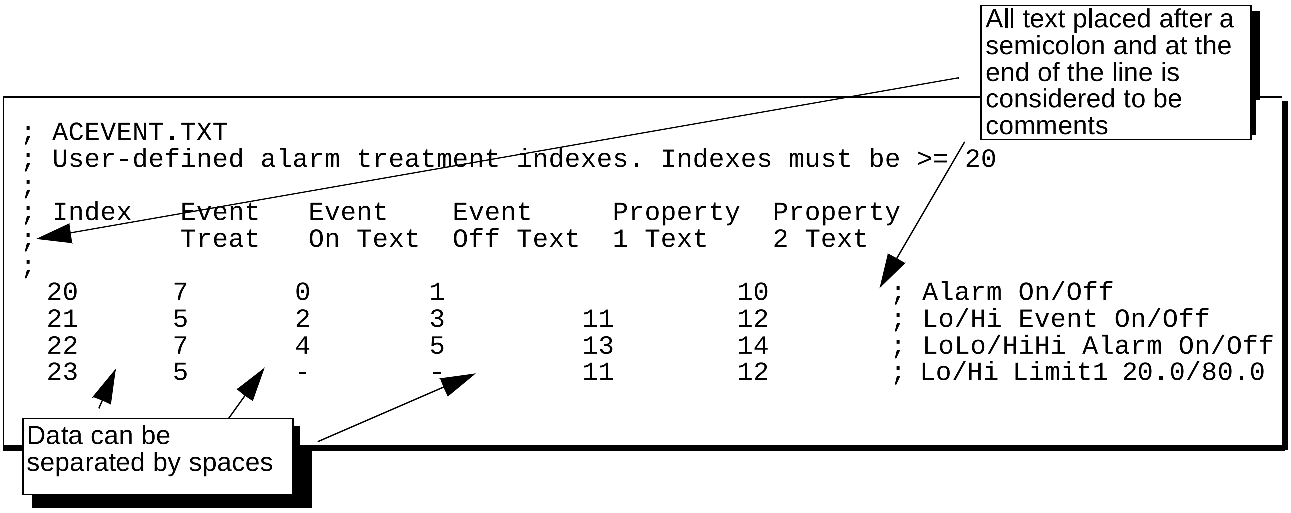
If, for example, an event is sent from a controller with the event reason EVENT ON, and the Event Treatment Word is set to 7, the event is treated as an alarm and is sent to all alarm destinations including Alarm Summary.

If an event with the event reason ALARM ON and the Event Treatment Word 1 is sent from a controller, the event is not treated as an alarm and sent to Alarm Summary, but it will be sent to all other alarm destinations. This means that if application specific Event Treatment Indexes are used, it is the Event Treatment Word that controls the actual event reason used in AdvaCommand.

- **Event On Text Number**: This is used when an object enters an event state. This number refers to a list of application specific defined texts that is read at start-up. The text should describe the new state that the object enters. For example, when a digital input signal enters an alarm state, the Event On Text could be ”Alarm”. To use the actual value of the object as Event On Text, use ‘-’ as Event On Text Number.
- **Event Off Text Number**: This is used when an object leaves an event state. This is a number referring to a list of application specific defined texts that is read at start-up. The text should describe the new state that the object enters. For example, if a digital input signal leaves an alarm state, the Event Off Text could be ”OK”. To use the actual value of the object as Event Off Text, use ‘-’ as Event Off Text Number.
- **Property 1 Text Number**: This is a number referring to a list of application specific defined texts that is read at start-up. The text should describe which property of the object has entered or left the event state, for example, a digital input value property could be ”Status”.

  If it is a Limit 1 or Limit 2 event, the Property 1 Text is used to specify the low limit property text, for example ”Lo Limit1”. For base objects, this text is used for the latched alarm status (A_LAALM) that is shown in alarm summary. The maximum text length that can be shown in alarm summary is six characters.
- **Property 2 Text Number**: This is a number referring to a list of application specific defined texts that is read at start-up. It is only used for Limit 1 and Limit 2 events, and specifies the property text to use for high limits, for example ”Hi Limit1”. For base objects, this text is used for the latched alarm status (A_LAALM) shown in Alarm Summary. The maximum text length that can be shown in alarm summary is six characters.

Following is an example of an application specific defined configuration file:



```
; ACEVENT.TXT
; User-defined alarm treatment indexes. Indexes must be >= 20
;
; Index   Event   Event    Event     Property  Property
;         Treat   On Text  Off Text  1 Text    2 Text
;
  20      7      0        1                 10        ; Alarm On/Off
  21      5      2        3        11       12        ; Lo/Hi Event On/Off
  22      7      4        5        13       14        ; LoLo/HiHi Alarm On/Off
  23      5      -        -        11       12        ; Lo/Hi Limit1 20.0/80.0
```

All text placed after a semicolon and at the end of the line is considered to be comments
Data can be separated by spaces

Figure 3-38. ACEVENT.TXT configuration file example

The example above, Figure 3-38, configures the following alarm/event texts:

- **The first row** configures alarm treatment for a digital signal. The signal generates alarms (to alarm summary and alarm printers). The alarm texts are ”Alarm On” and ”Alarm Off” (for event texts, see Figure 3-40).
- **The second row** configures Limit1 event treatment for an analog signal. The signal generates events (to alarm printers) when the value crosses the Limit1 low or high borders. The alarm texts are ”Hi Event On”, ”Hi Event Off”, ”Lo Event On” and ”Lo Event Off” (for event texts, see Figure 3-40, for property texts, see Figure 3-41).
- **The third row** configures Limit2 alarm treatment for an analog signal. The signal generates alarms (to alarm summary and alarm printers) when the value crosses the Limit2 low or high borders. The alarm texts are ”HiHi Alarm On”, ”HiHi Alarm Off”, ”LoLo Alarm On” and ”LoLo Alarm Off”.
- **The fourth row** configures Limit1 event treatment for an analog signal. The signal generates events (to alarm printers) when the value crosses the Limit1 low or high borders. The alarm texts are, for example, ”Lo 20.0 %” and ”Hi 80.0 %”. Note that the actual value of the object and the object's unit are used as event text. In this example, the low Limit1 is 20, the high Limit1 is 80 and the unit is ‘%’.

The alarm texts are used for all FIX alarm destinations except alarm summary, for example, alarm printers, alarm history, alarm network, etc. The format of alarm texts in Alarm Summary can be configured in FIX. The format of alarm texts to other alarm destinations, for example, the position of the different fields in the alarm text, **cannot** be configured. The format is as follows, see Figure 3-39.

```
          ”[a] [Object name] [Object description] [Property] [Event] [Time] [U]”
Position:   0   2             24                   54         64      82     104
```

Figure 3-39. Alarm text format for all alarm destinations except Alarm Summary

The fields have the following meanings, see Table 3-21.

Table 3-21. Alarm text fields

Field	Description
[a]	Shows the ‘*’ character if this is an alarm, is blank if it is an event
[Object name]	Name of the object that the event concerns
[Object description]	Description of the object that the event concerns
[Property]	The property of the object that the event concerns, for example, ”HiLim1”
[Event]	Describes the status of the property, for example, a value or ”On”
[Time]	The event time in the format: ”yy-mm-dd hh:mm:ss.ttt”
[U]	Shows a ‘U’ if the event time is uncertain, otherwise it is blank

The numbers in the **Event On Text** and **Event Off Text** columns, see Figure 3-38, refer to a file called **ACEVTEXT.TXT** located in the AdvaSoft directory, see Figure 3-40.

```
; ACEVTEXT.TXT
; User-defined Event On/Off texts, first index at 0
; Index   Text
;
  0       On
  1       Off
  2       ”Event On”
  3       ”Event Off”
  4       ”Alarm On”
  5       ”Alarm Off”
  ...
```

Figure 3-40. ACEVTEXT.TXT configuration file example

NOTE

Alarm texts intended for display in the Alarm Summary should be no longer than six characters.

The numbers in the **Property 1 Text** and **Property 2 Text** columns refer to a file called **ACPRTEXT.TXT** located in the AdvaSoft directory, see Figure 3-41.

```
; ACPRTEXT.TXT
; User-defined Event Property texts, first index at 0
;
; Index    Text
;
  …
  10       Alarm
  11       Lo
  12       Hi
  13       LoLo
  14       HiHi
  ...
```

Figure 3-41. ACPRTEXT.TXT configuration file example

3.5.18.3 Alarm line fields

The following fields are predefined as default in the alarm list:

Table 3-22. Predefined fields in the alarm list

Field	Length	Description
Tagname	21	Signal name. Name defined in the remote node. Configurator is displayed in this field.
Description	27	Description
Status	9	Alarm status. Can be one of the following: **Normal** **HiLim1** = High alarm **HiLim2** = High High alarm **LoLim1** = Low alarm **LoLim2** = Low Low alarm **CFN** = Change From Normal (digital alarm)
Value	12	Value when alarm occurs
Time In	12	Time when alarm occurs. Time stamp is sent from the remote node. Format: hh.mm.ss.ttt

NOTE

The **Time Last** and **Priority** fields, found in the **Column Format** dialog box, are not valid where distributed time stamping is used.

3.5.18.4 Changing display font size

The display font for the alarm display is **Courier new**. This font type cannot be changed. You can change the font size of Courier new by selecting the **Font!** command in Draw. Change the size as desired and click **OK** to accept changes.

3.5.18.5 Changing column order

You can rearrange column order in the alarm list as follows:

1. Open the alarm list display **_allist.odf**.
2. Double-click on the alarm list to load the **Alarm Summary Configuration** dialog box.
3. Open the **Column Format** dialog box.
4. Change column order by selecting the field you want to move in the **Column** order list, and move the field up or down with the **Up** and **Down** buttons on the right-hand side of the field list.
5. Click **OK** to accept the changes.

3.5.18.6 Creating separate lists for alarm areas

You may want to have separate lists for various process sections or separate lists for process and system messages.

To create your own list, follow these steps:

1. Open the alarm list display **_ALLIST.ODF**, and save it under a new name using the **Save As..**. command.
2. Edit the **Commands On Open** script for the new display.
3. Double-click on the alarm list to open the **Alarm Summary Configuration** dialog box.
4. Check that the **Filter** property is enabled.
5. Enable **Alarm area** and mark out or enable the areas you want in the list.
6. Save your display.

Table 3-23. Mapping between process sections and Alarm Areas

Process Section	Alarm Areas
0	All areas
1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	I
10	J
11	K
12	L
13	M
14	N
15	O
16	P

3.5.18.7 Configuration of the Historical trend data display

Historical Display

All of the display configuration is carried out in the Historical Display application found in the AdvaCommand program group.

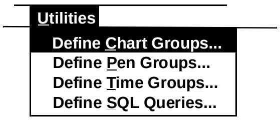
To carry out the configuration procedure, do as follows:

1. Create a pen group.

   During this process each process object is provided with its own pen, pen color and limits.

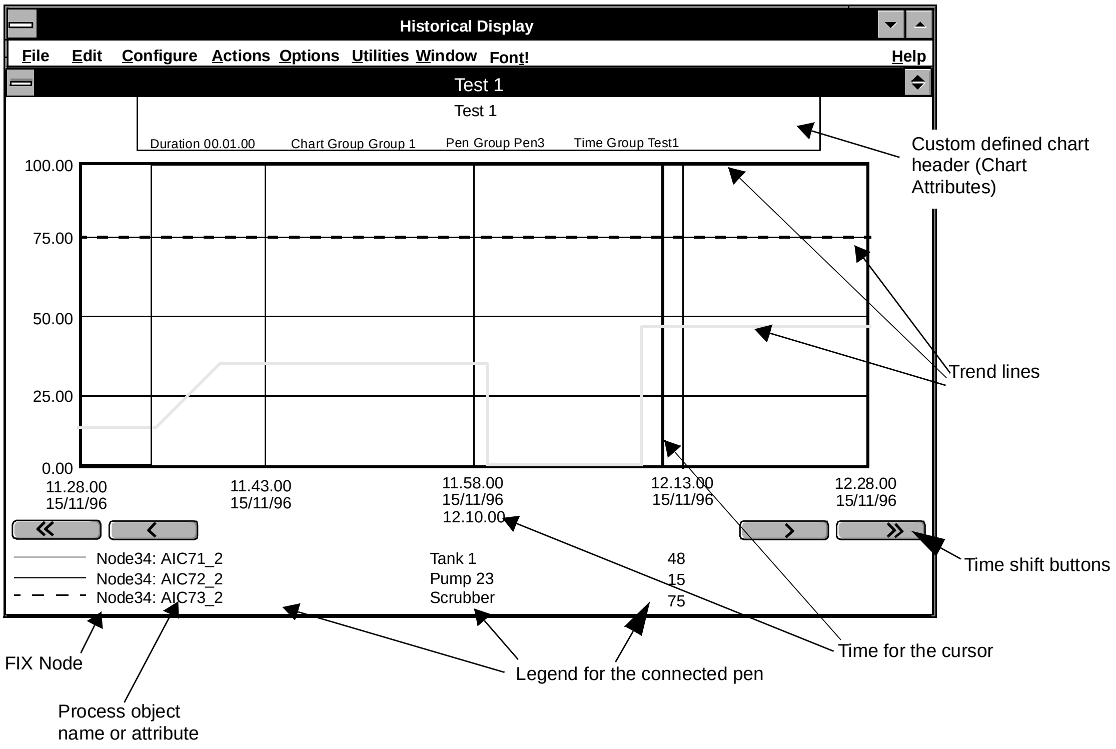
2. Create a time group.

   A time group defines start date, start time and time span.



3. Create a chart group.

   When you create a chart group, you assign the appropriate pen and time groups, created in the earlier steps, to the chart. You also set the attributes, such as header, coloring and legends, associated with the chart. For a more thorough description of the above operations, see the FIX documentation, the *Historical trending* section of the ***Advanced Tools*** volume.



Node34: AIC71_2	Tank 1	48
Node34: AIC72_2	Pump 23	15
Node34: AIC73_2	Scrubber	75

Figure 3-42. Chart display example

Also see the ***AdvaCommand Historical Trending User's Guide*** and the FIX ***Advanced Tools*** documentation for further information about Historical Assign, Collect and Display.

NOTE

In Historical Assign, the default subscription cycle time is nine seconds if no subscription rate is specified in the tagname independent of the collection rate specified in the dialog. For other cycle times, you must use the direct subscription, **'AI1;2S'.F_VALUE**. The various tag cyclic subscription times are S=seconds (default) and M=minutes. The collection rate should be equal to the tag cyclic subscription time.

NOTE

The tagname configured in HTA must match the tagname used in HTD exactly. This includes both the subscription specification described in the previous note and the field specification.

You must define trend data fetched via SQL in the HTD.INI file located in the default installed directory **.\LOCAL**. Also see the ***AdvaCommand Command Language User's Guide*** for further information.

NOTE

When you are making requests towards a SQL*Net database, the name and description attribute fields on the requested object must contain at least one character (ordinary character, space character, period character, etc.).

3.6 Configuration/Application Building Menus

See the FIX documentation for instructions on how to handle the FIX software. The *Draw* section found in the ***Display Development*** volume and the *Historical Trending* section found in the ***Advanced Tools*** volume are especially helpful.

Chapter 4 Runtime Operation

4.1 Product Operation

This chapter covers AdvaCommand for Windows procedures and predefined displays and functions which are used to monitor, supervise and control the process. For a description of how to perform these tasks, see the ***View*** manual. Refer to *Section 1.5.1, AdvaCommand for Windows Documentation Binders* for references to manuals describing advanced topics such as alarming and trends and reports.

When you start AdvaCommand for Windows, it initializes and starts underlying supplied software and hardware, including AdvaSoft for Windows and communication interfaces.

To enable exchange of data with the automation system, the communication network and remote nodes must be configured and running.

Process values update with the cycle time specified for each process value. Alarms are written to the alarm list and include the local time stamp generated in each remote controller.

4.2 Operating Overview

When AdvaCommand for Windows is started, monitoring and control are operational when the automation system is connected and running. The full functionality, as summarized below, is available to the operator on predefined and custom displays. The operational functionality included in AdvaCommand runtime is summarized below.

Monitoring is the display of real-time process data to the operator through various display types. The predefined displays include group, object, alarm and event. Additional plant-specific process and trend displays may be included. A dialog is enabled for each display through the dynamic function keys. You can access the dialog using the mouse and keyboard. Additional fixed display selection keys are pre-programmed for quick selection of important and frequently used displays.

Supervisory control, the ability to simultaneously update process data to elements on the screen while issuing commands to the object, is achieved by selecting objects and issuing commands using the mouse on the dynamic function keys or through custom fixed function keys. When you select objects, the ABB logo is replaced by an object close-up displaying more information. At the same time, you activate the dialog for the object through the dynamic function keys accessible with the mouse or keyboard function keys.

Alarms and events are shown in lists and in the display header. The last process alarm is always displayed at the top of the operator screen in the display header. System and process alarms and events are displayed in dedicated lists accessible from the **Display** menu or through function keys when you select the alarm line. You can call up alarm lists directly and acknowledge alarms using the pre-programmed keyboard keys or mouse.

Data archiving and trending is sampling and storage of data in files at rates specified by the application engineer or operator. You can display the data in trend displays and use it in reports.

Reports, if included for your plant, are often used for administrative purposes and contain calculations and information such as plant performance. Reports can be defined during engineering of the plant or by the operator. You can use spreadsheets such as Microsoft Excel for reporting. A set of macros are included for use with Microsoft Excel.

4.2.1 Display areas and objects

The figure below shows the display area layout and a process display example with display objects for monitoring and supervisory control.

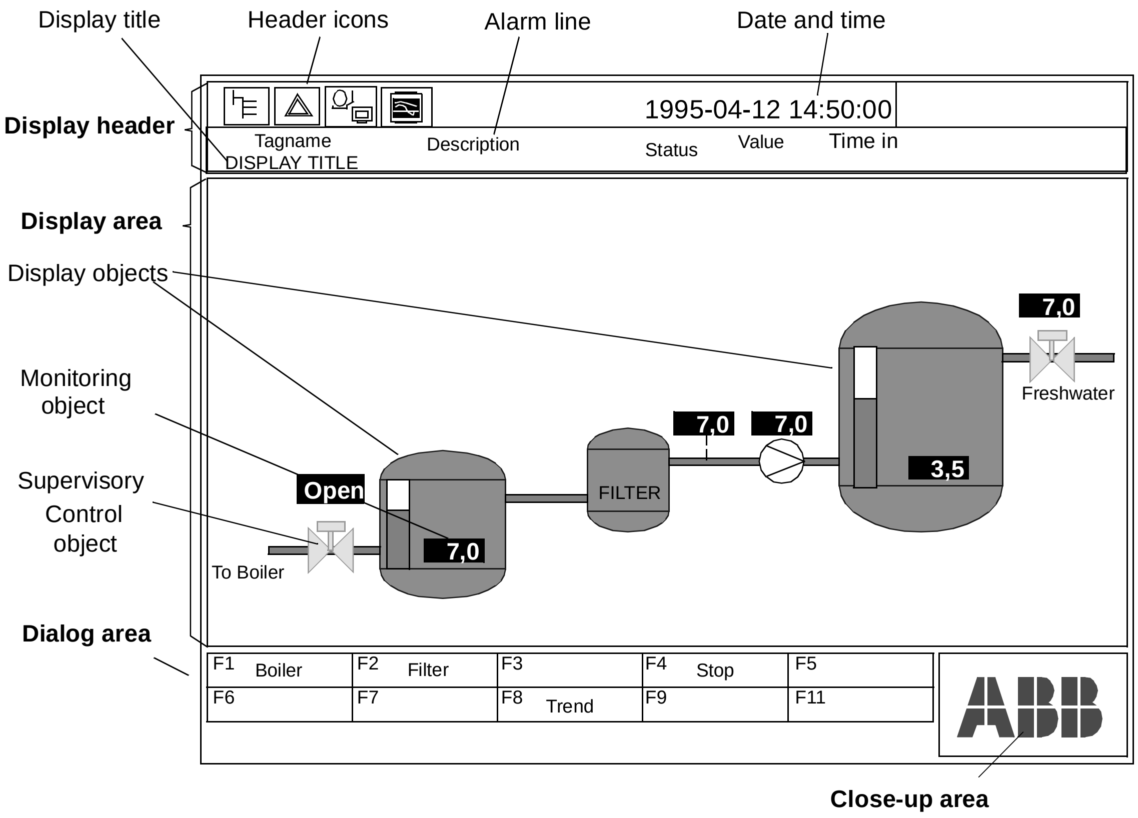


Figure 4-1. Process display showing example

The standard display layout includes the **display header**, **display area**, **dialog area** and **close-up area**. The dialog area and object close-up area are dynamic and activated depending on the current display and selected objects.

- When you select an object, information about the object is presented in the **object close-up area**.
- The dialog for the current display or selected object is assigned to the **dynamic function keys** in the **dialog area**.

4.2.1.1 Screen header

The screen header is present on all displays and contains the following elements. Refer to Section 4.4, Operating Instructions for operating details.

Display Menu icon

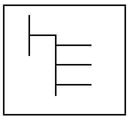


Figure 4-2. Display menu icon

The display menu icon gives you direct access to the **Display** menu.

Alarm List icon

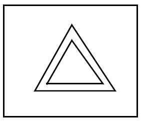


Figure 4-3. Alarm list icon

The alarm list icon gives you direct access to the alarm list.

Alarm History icon

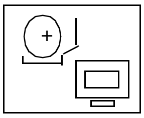


Figure 4-4. Alarm history icon

The alarm history icon gives you direct access to the Alarm History application.

Historical Trend Display icon

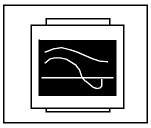


Figure 4-5. Historical trend display icon

The historical trend display icon gives you direct access to the Historical Trend Display application.

Date and time display

1995-04-08 13:58:00

Figure 4-6. Date and time display

The date and time display shows the current date and time on the system clock.

Alarm Line

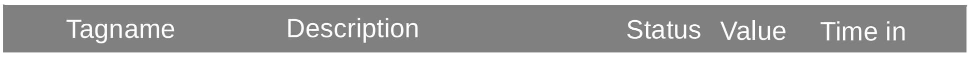


Figure 4-7. Default alarm line layout

The **last unacknowledged alarm** or acknowledged alarm still in alarm status is always displayed in the alarm line. You can acknowledge the alarm in the alarm list or you can select the object on a process display and acknowledge it with the acknowledge key.

Display Title

The title of the current display is shown directly under the alarm line.

4.2.1.2 Display area

The display area can contain a plant-specific process display or any of the other standard displays and lists discussed later in this section.

4.2.1.3 Dialog area

F1	F2	F3	F4	F5
F6	F7	F8	F9	F11

Figure 4-8. Dynamic Function keys

The **dialog area** contains the **dynamic function keys**, which change function depending on the **current display** and **currently selected object**. The object dialog is activated when you select an object from a process display.

4.2.1.4 Object close-up area

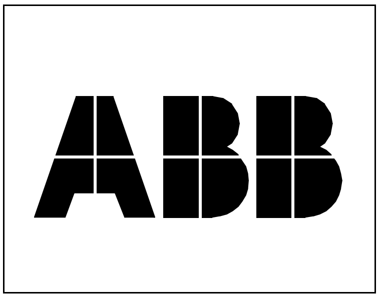


Figure 4-9. Object close-up area when no object is selected

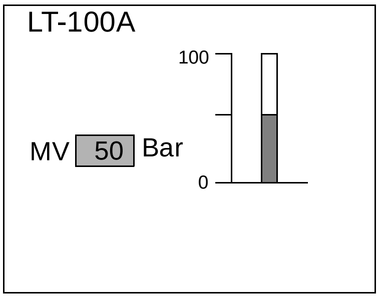


Figure 4-10. Object close-up area showing selected object

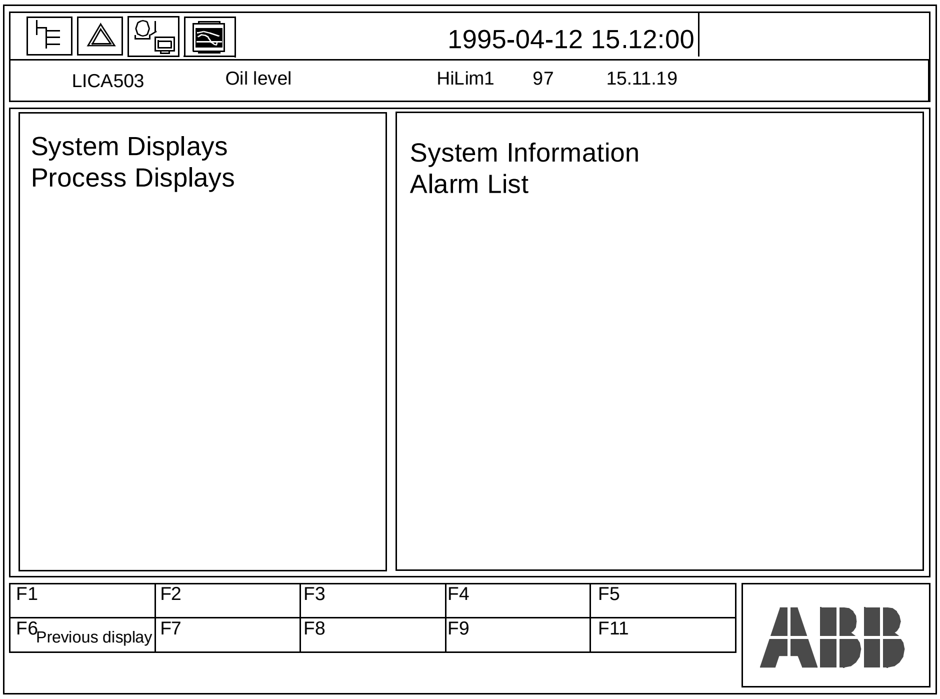
The **Object close-up** replaces the ABB logo when you **select** an **object** on a process display. It shows more information about the currently selected object.

4.2.2 Predefined display types

The predefined displays include the layout and operations to call up the displays, including dynamic function keys. The layout and operation summary follows. For a detailed description of how to operate these displays, see Section 4.4, Operating Instructions.

4.2.2.1 Display menus

The predefined **Display** menu consists of two display parts, the display group and display menu list.


1995-04-12 15.12:00

LICA503 Oil level HiLim1 97 15.11.19

System Displays
Process Displays

System Information
Alarm List

Figure 4-11. Display menu display

4.2.2.2 Group displays

The object group displays have a predefined layout and dynamic fields, as shown below, and include up to 10 object close-up displays.

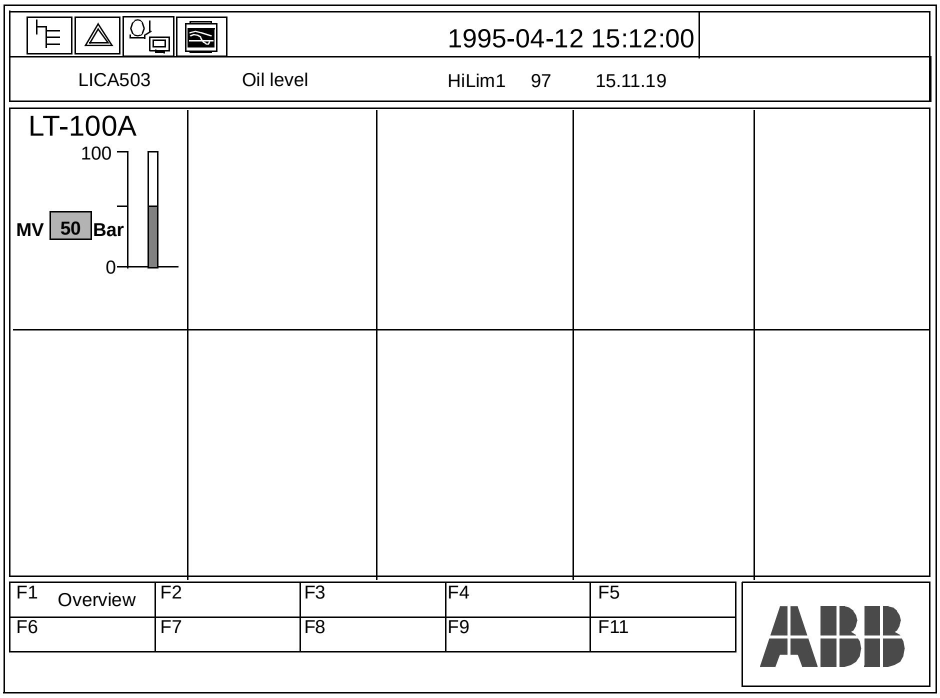


Figure 4-12. Group display layout with examples

4.2.2.3 Object displays

The predefined object displays include static and dynamic properties for the basic objects AI, AO, DI and DO.

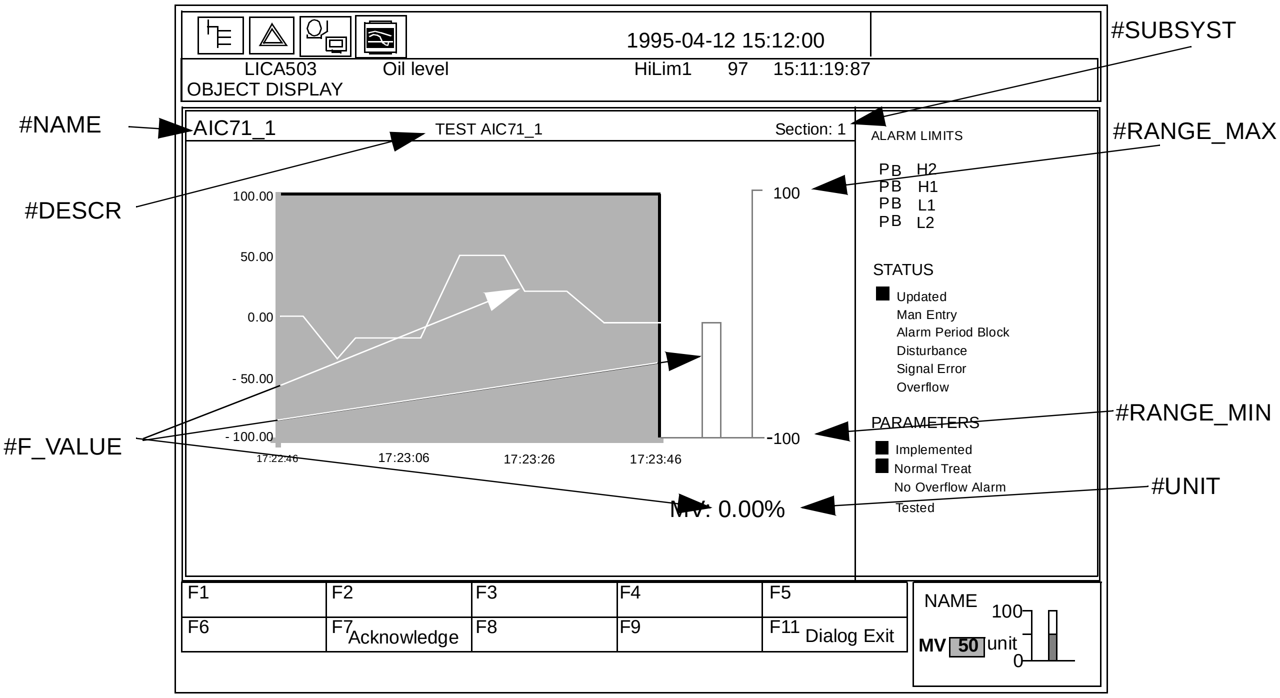
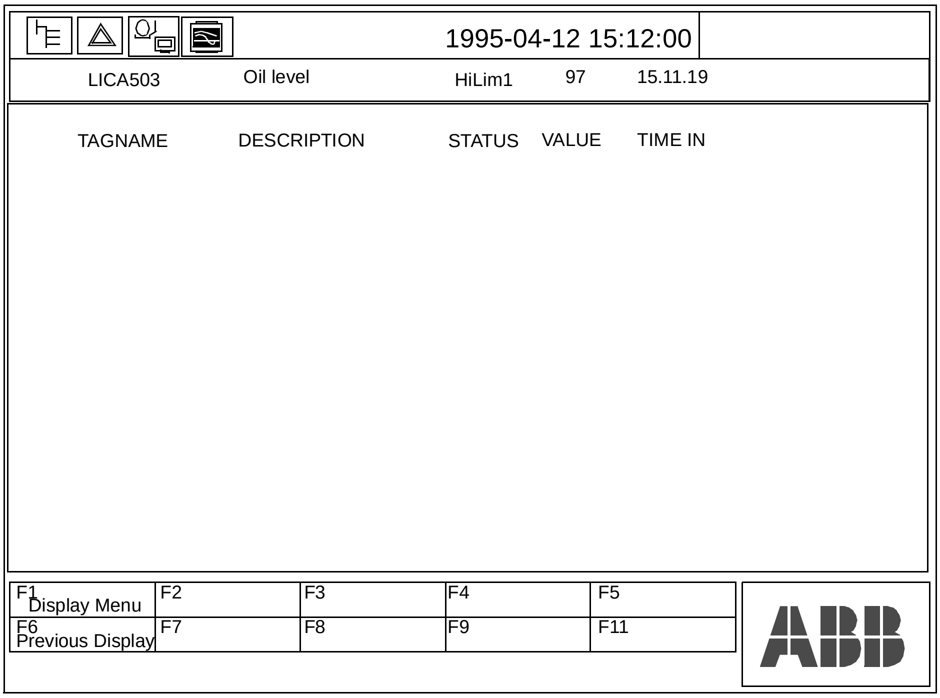


Figure 4-13. Analog object display layout

4.2.2.4 Alarm and event lists

The alarm summary display shows system alarms and process alarms and events in chronological order, with the time stamp generated locally or in the operator station. The default alarm line layout is shown below. The layout and fields included are user defined and may, therefore, differ from the default shown below.



1995-04-12 15:12:00

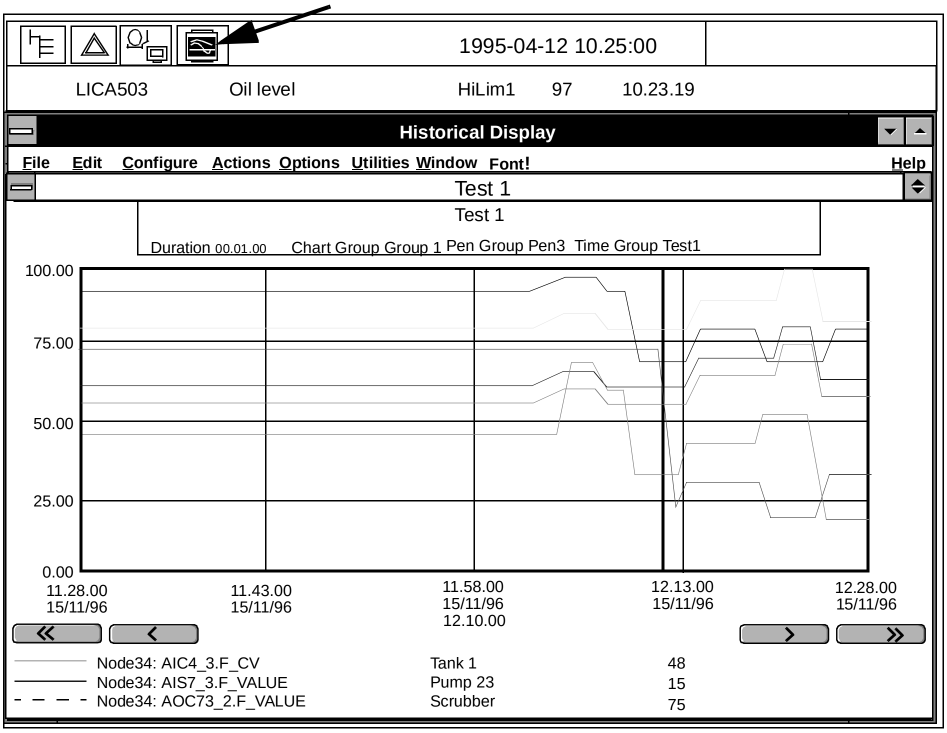
LICA503	Oil level	HiLim1	97	15.11.19

TAGNAME	DESCRIPTION	STATUS	VALUE	TIME IN

Figure 4-14. Alarm summary display

4.2.2.5 Trend displays

Trend displays show archived data in graphic form. An example using the default trend display layout is shown in the figure below.



1995-04-12 10.25:00

LICA503 Oil level HiLim1 97 10.23.19

Test 1

Duration 00.01.00 Chart Group Group 1 Pen Group Pen3 Time Group Test1

Node34: AIC4_3.F_CV	Tank 1	48
Node34: AIS7_3.F_VALUE	Pump 23	15
Node34: AOC73_2.F_VALUE	Scrubber	75

Figure 4-15. Example of a trend display

To start the Historical Trend Display, click on the HTD icon.

NOTE

HTD is a separate application and is not part of View.

4.3 Runtime Tutorial

Whether you are a new or experienced user, it is recommended that you go through the QuickStart Tutorial when you are using a new version of AdvaCommand Runtime and Development. It includes version news and application-building and run-time operation examples.

4.4 Operating Instructions

Most PCs are equipped with a standard keyboard and a mouse. This guide tells you how to operate using a standard **alphanumeric keyboard** with **dynamic** and **fixed function keys**, the **pointing device** (mouse, track ball or touch screen supported) and the **keyboard function keys**.

You can perform most AdvaCommand for Windows runtime functions with a standard alphanumeric keyboard, although a pointing device used with the keyboard is recommended for ease and speed of operation.

4.4.1 Keyboard operation

In the figure below, you can see a standard alphanumeric keyboard with the **dynamic function keys** on top and a number of predefined **fixed function keys**. In addition, there are a number of predefined functions using the **<Ctrl.>** key, as described below.

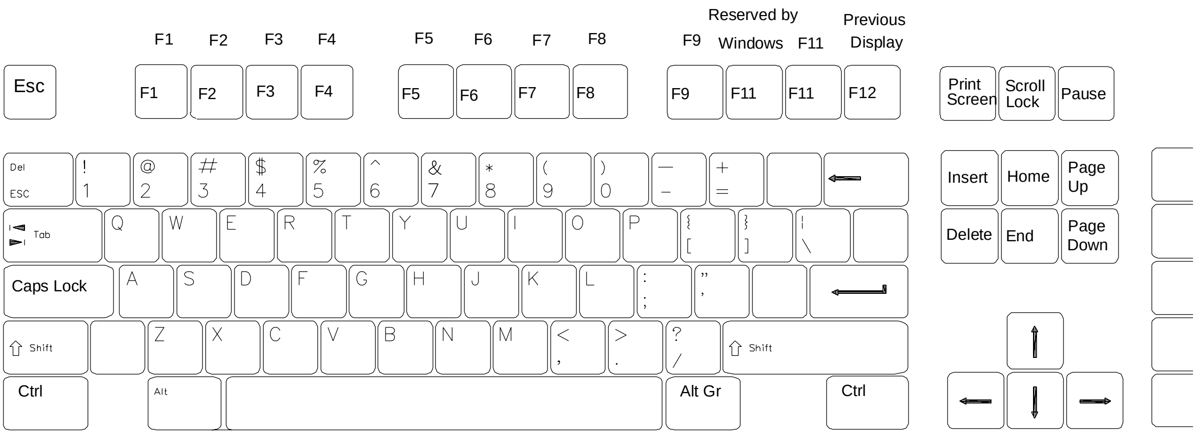


Figure 4-16. Alphanumeric keyboard with keyboard function keys

4.4.1.1 Fixed keyboard keys

You can assign **default predefined** and **plant-specific** functions to **fixed keyboard keys** or to a **combination of keystrokes**. These keys are independent of which displays are open or which dialog is active. They perform operations such as selecting certain displays or commands to process objects. The default (not plant specific) keyboard function keys are as shown in the table below.

Table 4-1. Default keyboard function keys in View.kmx

Key(s)	Function
<Ctrl-C>	Menu **Close**
<Ctrl-D>	Menu **Fit_In_Window**
<Ctrl-H>	Menu **Silence_Horn**
<Ctrl-O>	Menu **Open**
<Ctrl-R>	Menu **Toggle_Horn**
<Ctrl-Y>	Menu **Zoom_To**

4.4.1.2 Dynamic keyboard keys

You can access the **current dialog** (functions assigned to the **dynamic function keys F1 - F11**) with the pointing device or with the default **dynamic keyboard keys F1 through F11** (although you can override the default functions if plant-specific conditions require). There are two types of dialogs, the display dialog and the object dialog. The **display dialog** is active for the **current display** if no objects are selected. The **object dialog** is activated when an **object is selected**.

4.4.2 Operating with a mouse

Plant operation is most efficient when you use a pointing device together with the keyboard. The mouse is the most widely used pointing device, and runtime operations are described for a standard Microsoft **Windows compatible** mouse with a **left select button**. Touch screens and other pointing devices emulating standard Windows compatible mouse functionality, as specified in the *Environment* manual, work in the same manner.

4.4.3 Choosing displays

You can select displays from the **Display** menu, dynamic function keys, keyboard function keys or display links. Select:

- From a **Display** menu using cursor positioning.
- By customer-defined **Direct Selection** keys.
- By the **Dynamic Function Keys.**
- By **Display Links** in process displays.

4.4.3.1 Selection from the display menu

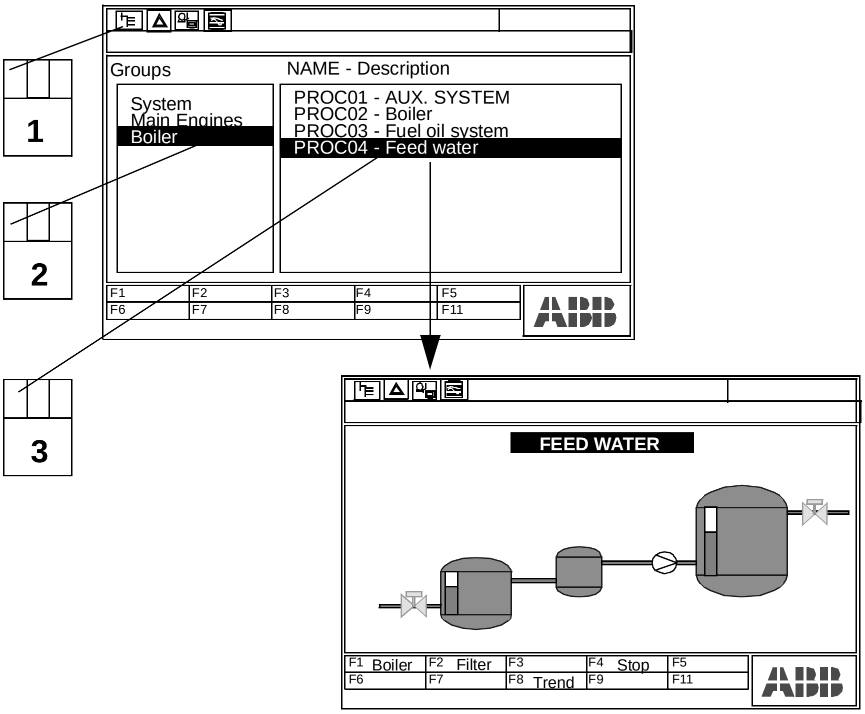


Figure 4-17. Display selected from a Display menu

The displays are listed in groups. The displays can be base, overlap or both.

Base displays are displays that cover all of the display area. Overlap displays cover a part of the display area and are conditional; that is, they appear when, for example, you select an object.

1. Open the **Display** menu by clicking the mouse cursor on the **display menu icon** from the display header or by pressing **F1.**

   A list with all groups is displayed.

2. Position the cursor on the group name and click the **Select** mouse button.

   The displays stored in the groups are listed.

3. Position the cursor on the display name in the list and click to call up the display.

4.4.3.2 Direct display selection

You can perform direct display selection in two ways:

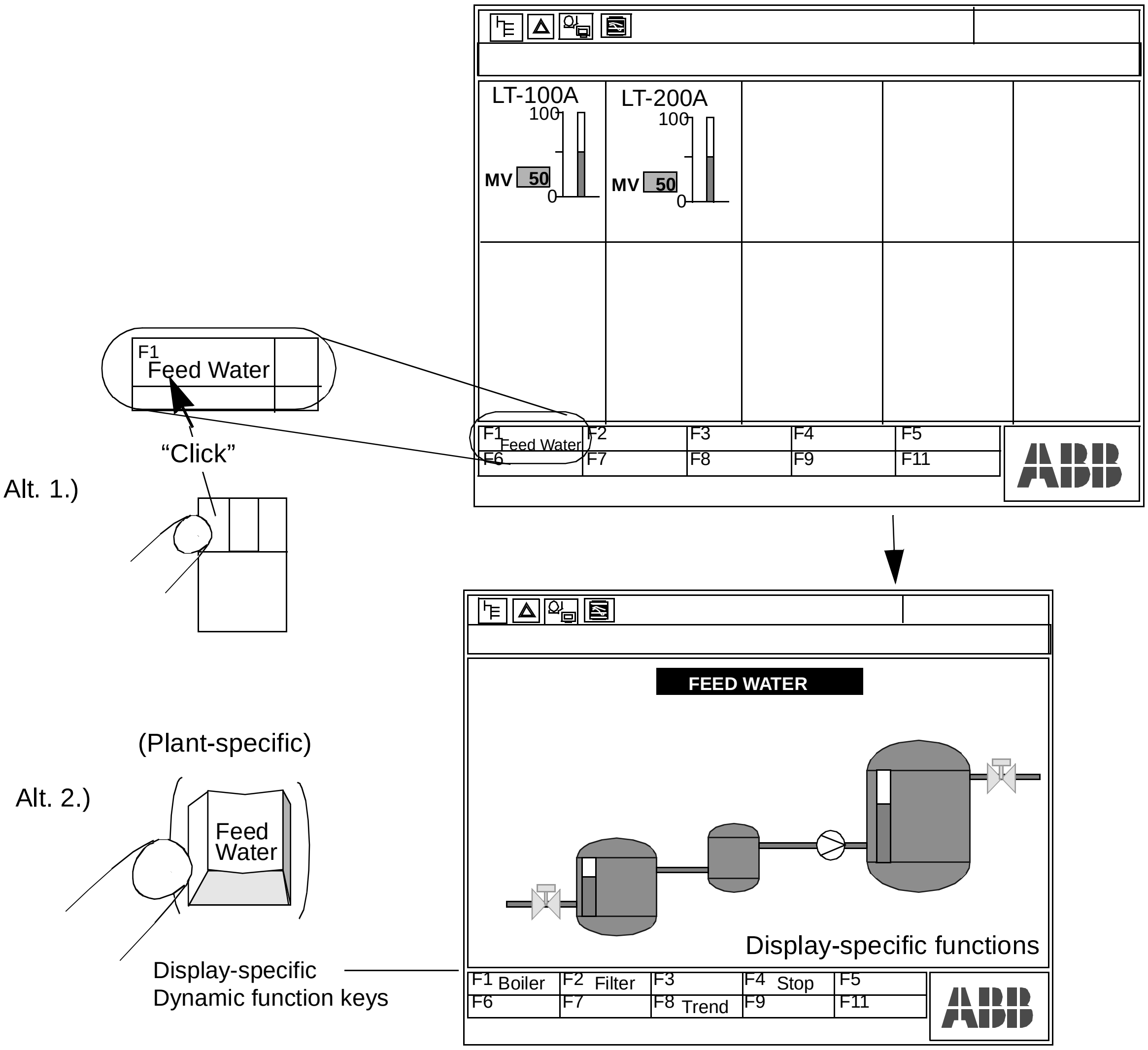


Figure 4-18. Display selection with the dynamic function keys

1. To choose displays using the dialog, click the mouse on the corresponding dynamic function key.
2. To choose displays with a **plant-specific keyboard button**, press the required key from any display.

4.4.3.3 Previous display selection

You can toggle between the current display and previous display in three ways:

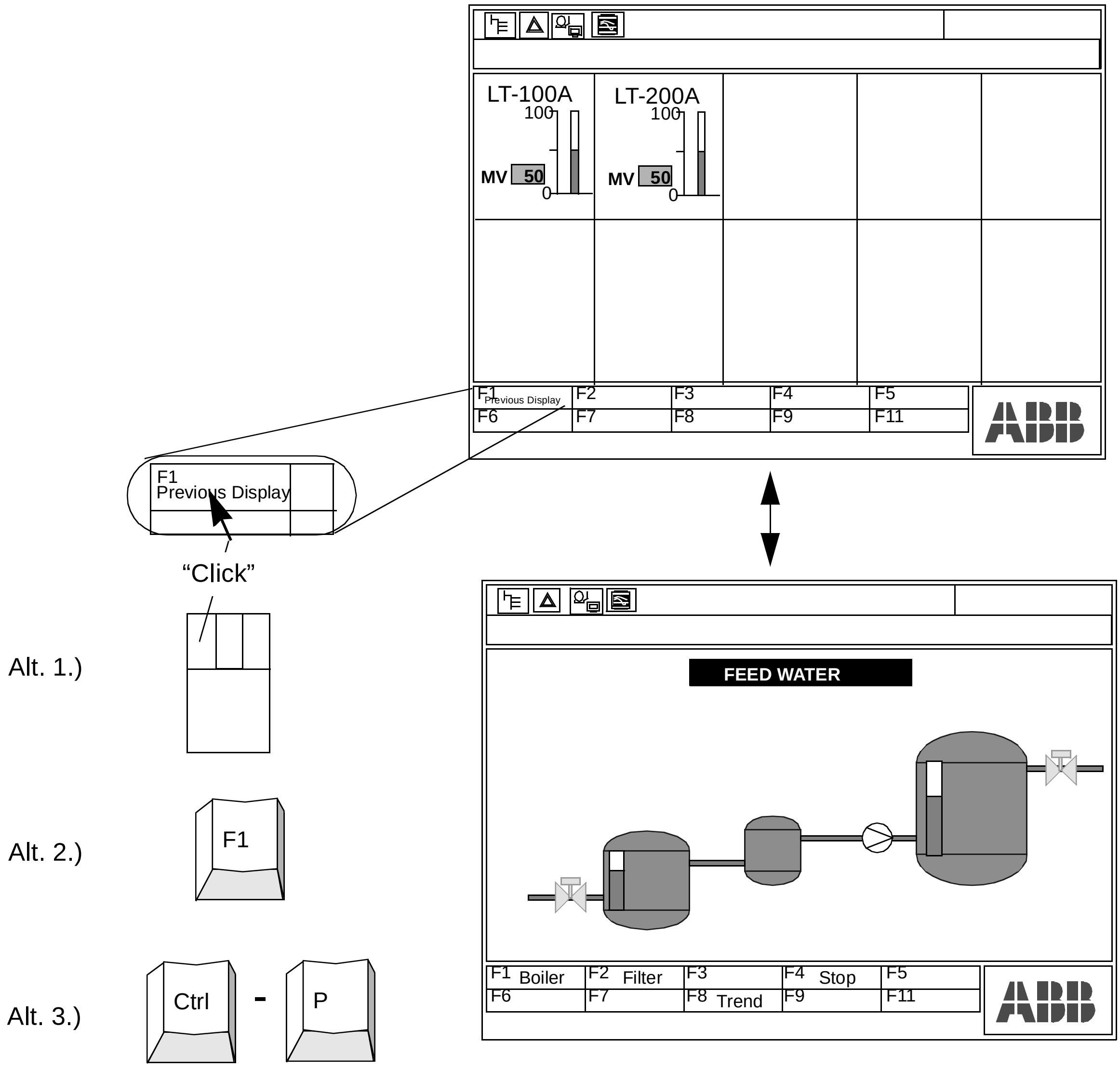


Figure 4-19. Display selection with the "Previous Display" key

1. Click on the **Previous Display dynamic function key** with the **mouse cursor**.
2. Press the keyboard function key corresponding to the dynamic function key.
3. Click **<Ctrl.-P>** from any display.

4.4.3.4 Paging

Displays can be **linked** together. Use paging to page through a list or a display that consists of several pages. You can page in one of three ways:

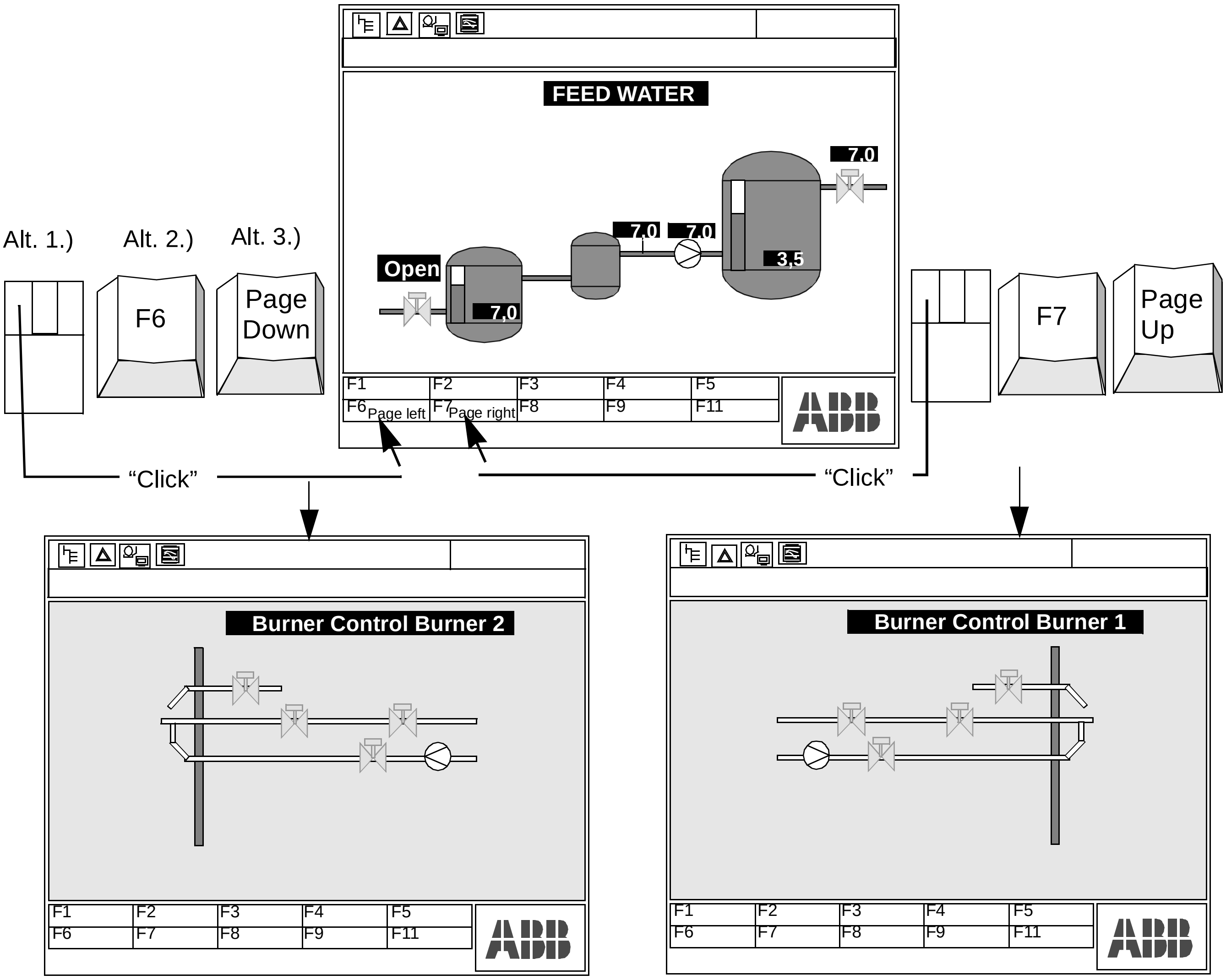


Figure 4-20. Display selection with paging keys

1. To page between linked displays, click on the **Previous page dynamic function key** with the mouse or
2. Press the corresponding **keyboard function key** or
3. Press the **Page Up** or **Page Down** key, which are independent of active dynamic function keys and the display.

4.4.4 Monitoring and supervisory control

The **display object**, normally an image of a **process object**, includes all object information, called attributes. An object can be an **I/O point** with object attributes like alarm limits and status information. It can also be a complicated control loop known in the controller as a **type circuit**, like an on/off motor control or PID regulator.

You can **monitor** information for specific objects by **selecting** objects and observing object information from the process and close-up display and the object display.

You can perform **supervisory control** by **selecting** a display object and/or **controlling** the object via the dynamic function keys or fixed function keys.

To close the object close-up display and deactivate the dialog and control over a selected object, **de-select** the object.

4.4.4.1 Selecting a process object

Alternatives for selecting an on/off valve controlled by a single digital output are described below.

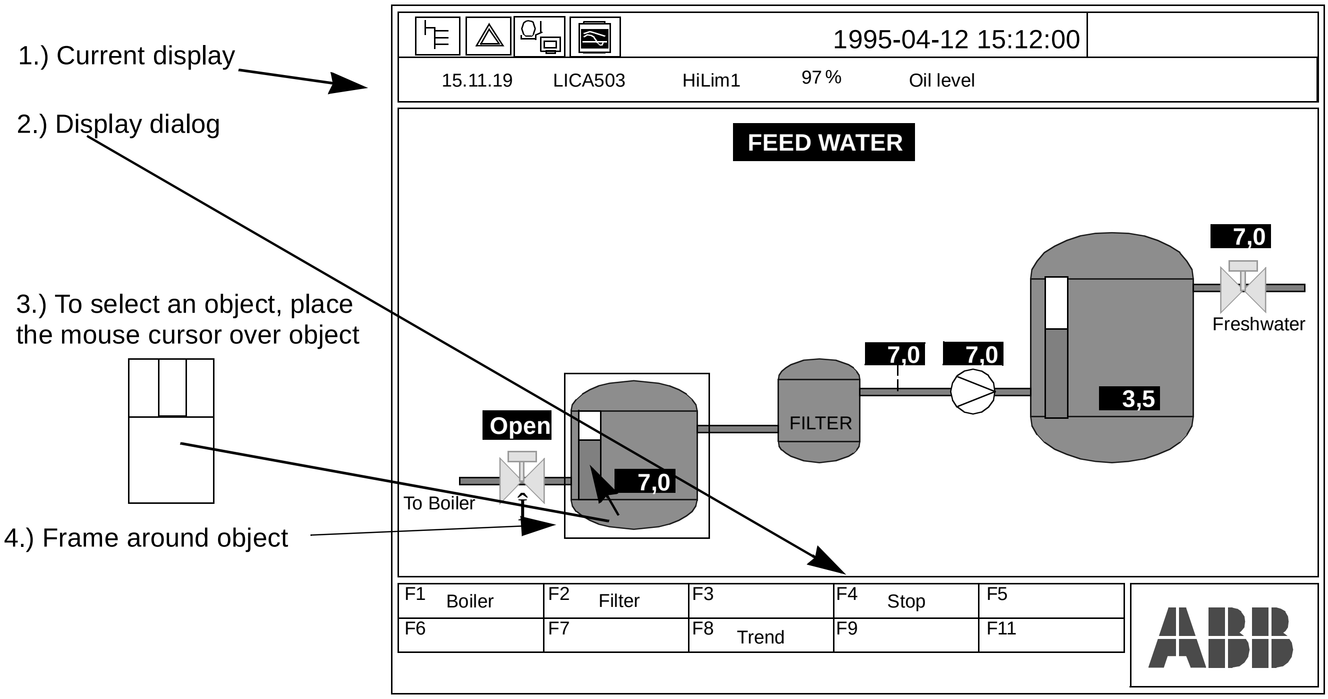


Figure 4-21. Cursor position on the object

You can select objects in the following ways:

1. Choose a **group display** or a **process display**, like in this example, containing the object you wish to select.
2. The dialog for the display is presented.
3. Position the cursor on the **object name** in a group display or on the **object symbol** in a process display.
4. A single-line frame appears around the object, to indicate that the cursor is in the object area.

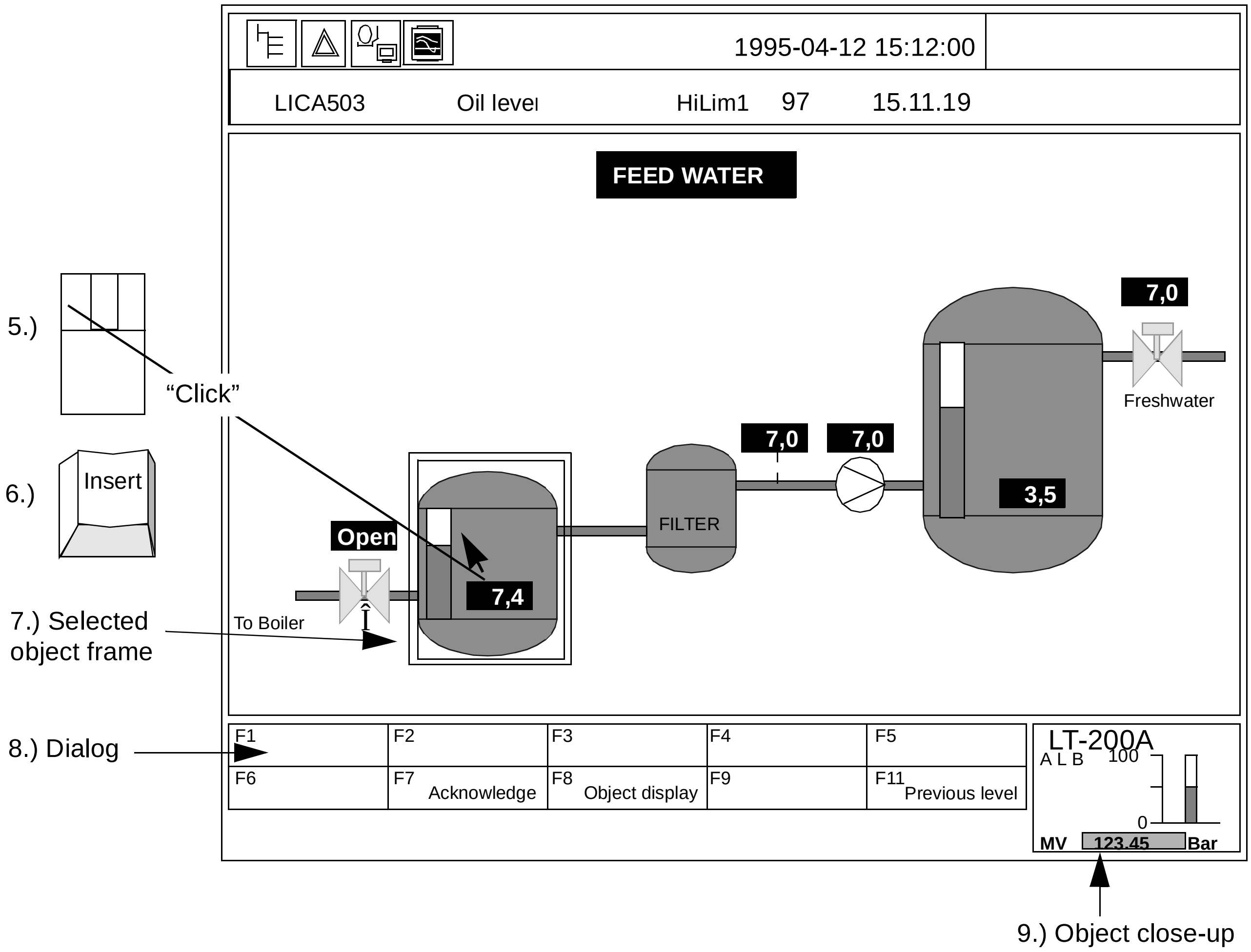


Figure 4-22. Selection of the object

5. Click the **Select** mouse button or
6. Press the **Select keyboard key**, assigned to the **Insert** key by default.
7. The object symbol selection is indicated by a **double-line frame** around the object.
8. The active **dialog** for the object is presented.
9. A **close-up** display of the object is displayed in the object close-up area.
10. To reset selection of objects, click the **Deselect** keyboard key to de-activate control over the object and regain the display dialog.

4.4.4.2 Monitoring a process object

Object displays for standard and custom objects are provided with your system. Standard objects are analog input (AI), analog output (AO), digital input (DI) and digital output (DO). Object displays show dynamic information, give you control of the currently selected object and include displays as shown below.

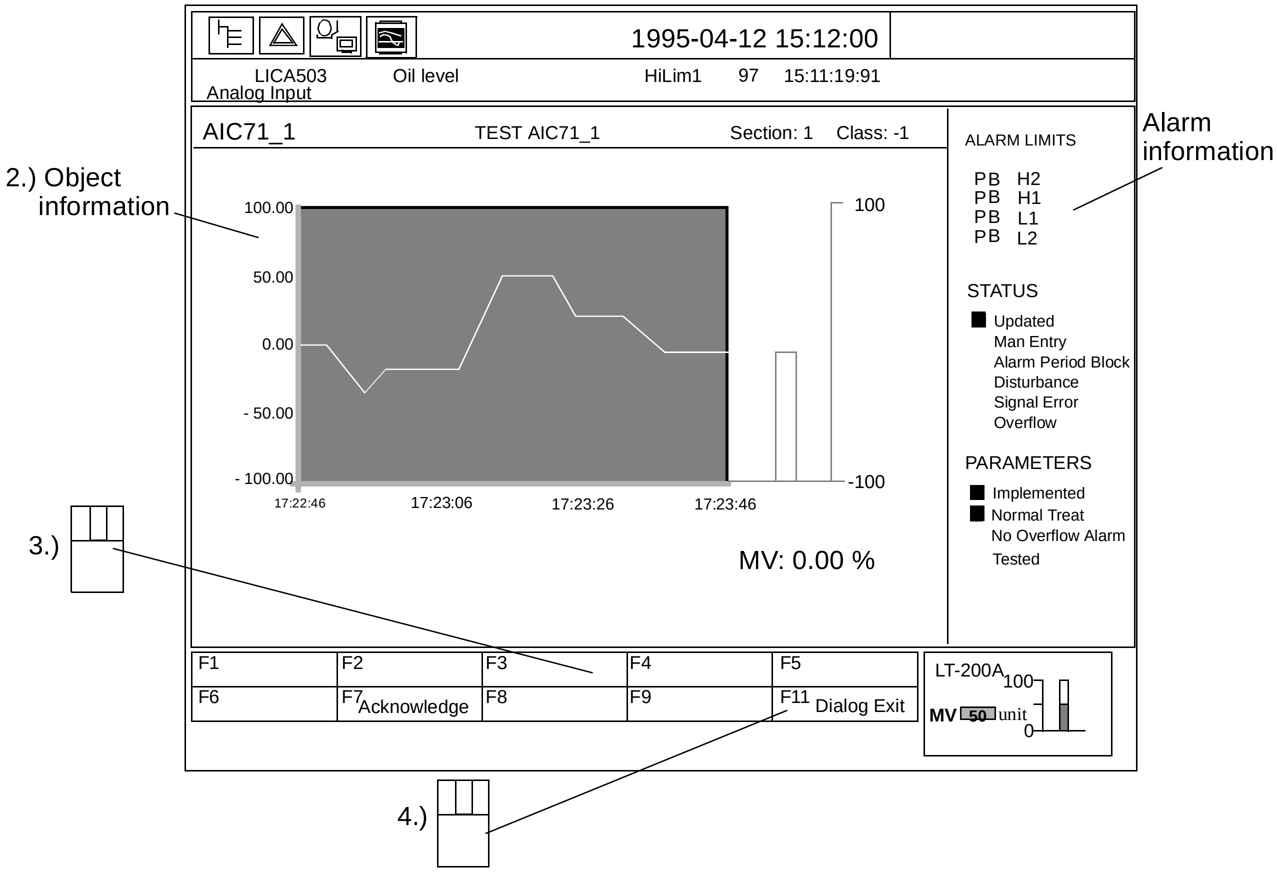


Figure 4-23. Object display for selected object

1. Select an object on a process display and choose the function key to call up an object display.
2. The object display for the object type appears with dynamic values and information.
3. Change **object parameters** by clicking the dynamic function keys 1% or 10% until expected value (only for AO).
4. Return to the process display retaining object selection by choosing **dialog exit**.

4.4.4.3 Controlling process objects

You can control digital objects like on/off and open/close type functions to digital objects from **process displays** or **object displays**. An example of control from a plant-specific process display is described below.

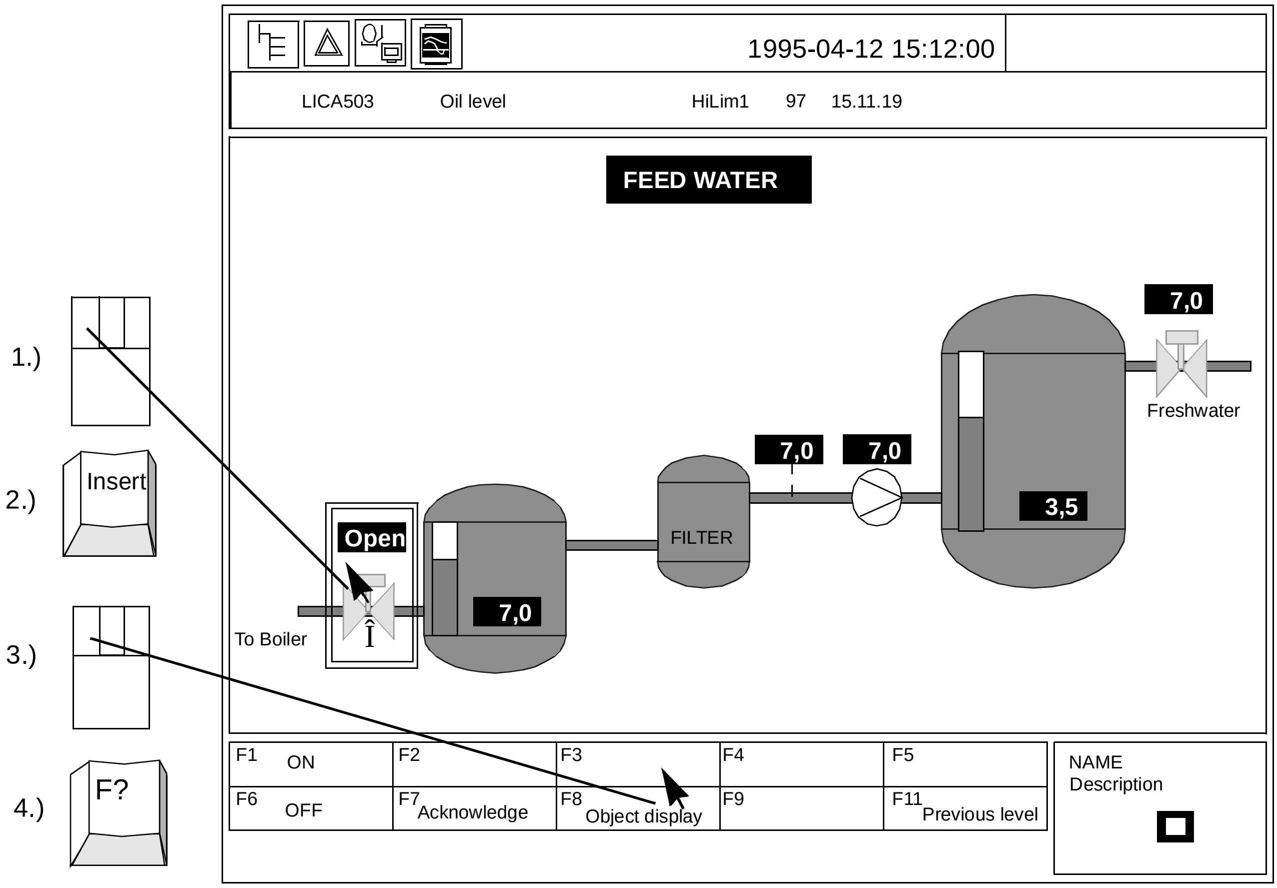


Figure 4-24. Selection of the object

1. Select a digital object by **clicking** on the **chosen object** with the mouse cursor or
2. By pressing the **Select keyboard key**, the **Insert** key by default.
3. **Send commands** to the object by clicking on the **dynamic function keys** on the screen or
4. By pressing the corresponding **keyboard function keys**.
   If the command is successful, the value changes to the new current value of the object.

4.4.4.4 Process control from object displays

To get the maximum supervision of the process object when controlling it, you can choose the object display for the currently selected object in a process display in two ways:

1. Click on the **Object display dynamic function key** or
2. Press the corresponding keyboard function key (if defined).

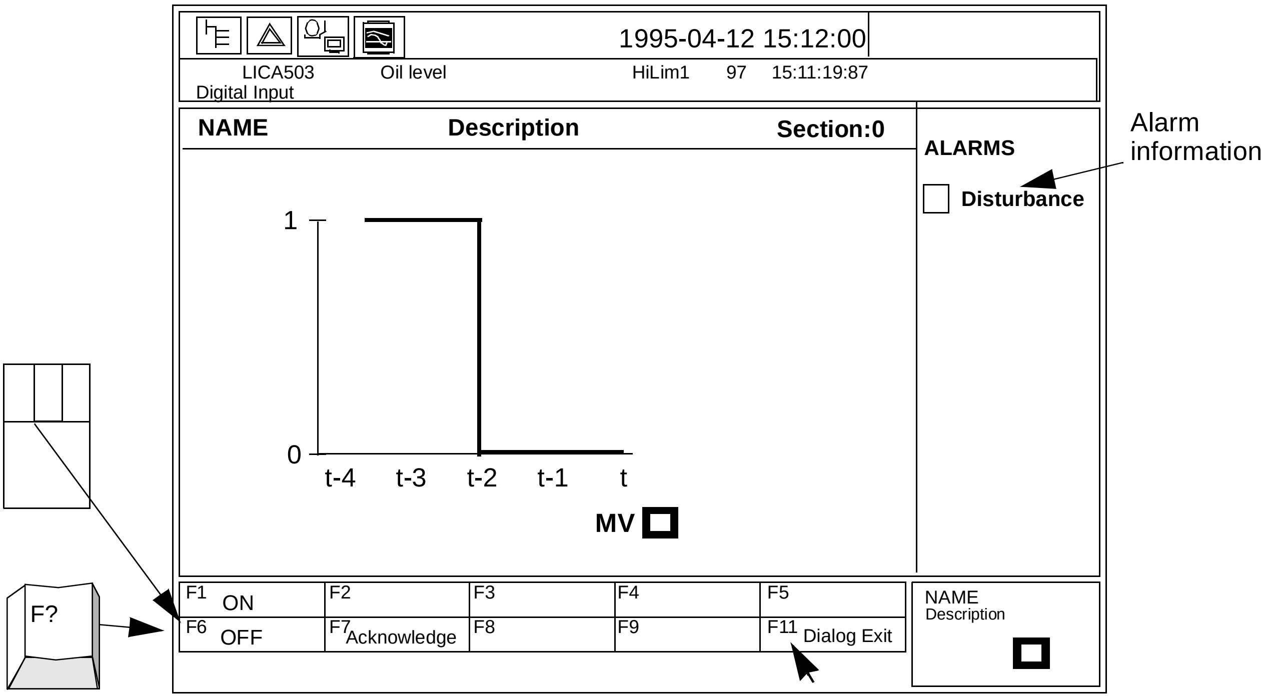


Figure 4-25. Control of a digital object from the object display

The dialog for the selected object is retained, and control is the same as from the process display.

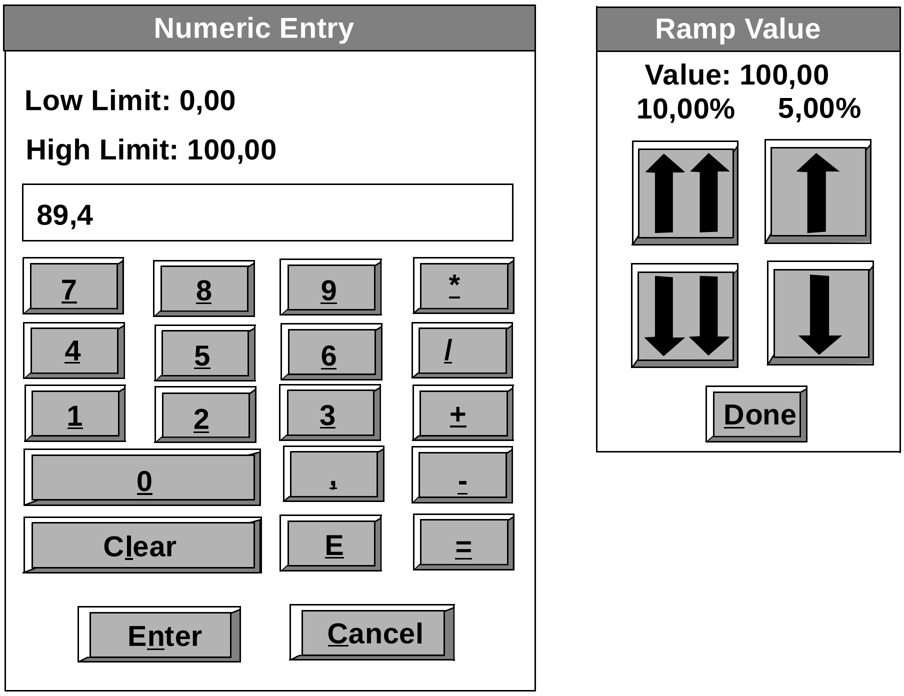
4.4.4.5 Controlling analog objects

Analog can be one of the following types:

- Alarm limits for measured values, output signals and set points
- Set points
- Manual output signals
- Control parameters.

Analog values are entered into the system in one of two methods:

- with a numeric keypad
- with a ramp function.



Numeric Entry

Low Limit: 0,00

High Limit: 100,00

89,4

Ramp Value

Value: 100,00

10,00% 5,00%

Figure 4-26. “Calculator” for numeric and Ramp function dialogs for data entry.

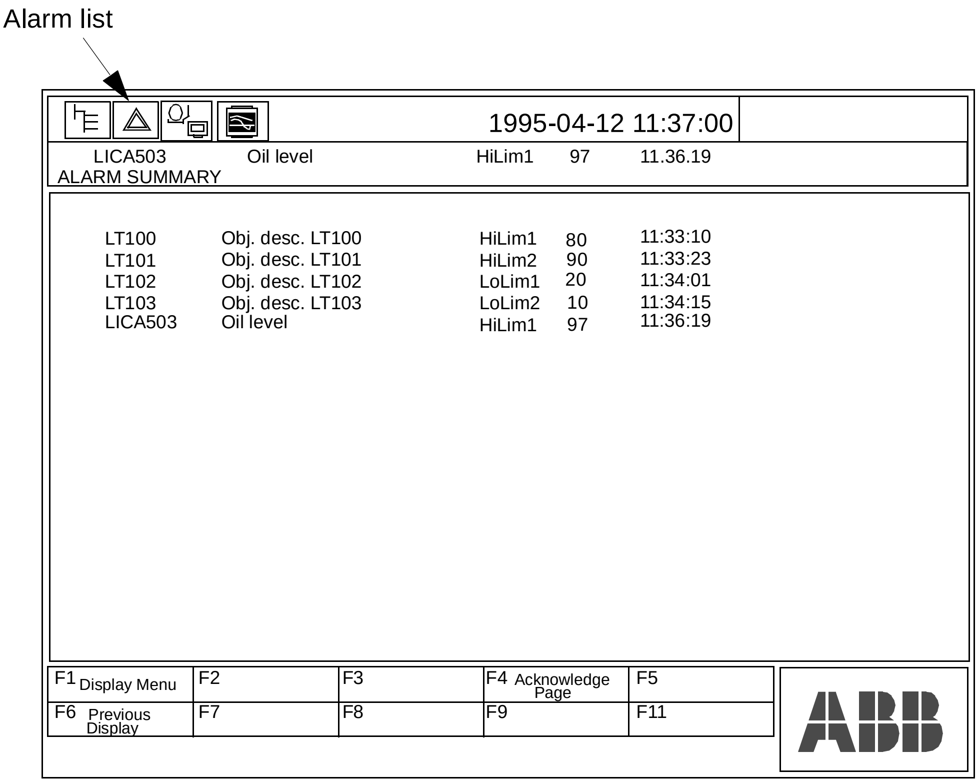
4.4.5 Event and alarm handling

Event and alarm messages are stored and presented in **chronological lists**:

- Process alarm list
- Process event list.

Process alarms are listed in the **alarm list**, which is also called the Alarm Summary and header display. An alarm is an event with a normal position and a not-normal position. It is **acknowledged by the operator**. Alarms are removed from the list and display header when the operator has acknowledged the alarm and the error state has returned to normal.

The default process alarm list and layout are shown below. The plant-specific alarm line layout and dialog may differ from the default.


Alarm list

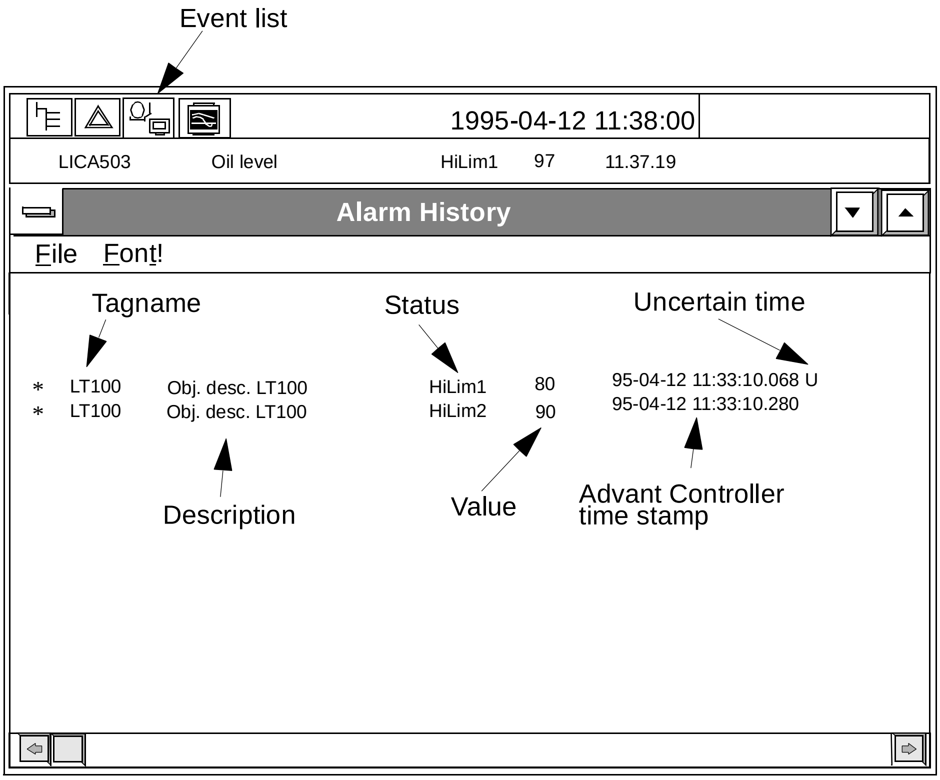
```
                                                   1995-04-12 11:37:00
   LICA503          Oil level            HiLim1     97       11.36.19
ALARM SUMMARY

    LT100       Obj. desc. LT100         HiLim1     80       11:33:10
    LT101       Obj. desc. LT101         HiLim2     90       11:33:23
    LT102       Obj. desc. LT102         LoLim1     20       11:34:01
    LT103       Obj. desc. LT103         LoLim2     10       11:34:15
    LICA503     Oil level                HiLim1     97       11:36:19
```

Figure 4-27. Alarm Summary Layout

Choose the alarm list display from the icons in the upper left-hand corner.

Process events include alarms and events and are listed in the **event list**, also called the Alarm History. Events happen when process values, such as digital values and analog value limits, shift state. The time is recorded and sent to the event list in the operator station.

The default process event list and layout are shown below.



1995-04-12 11:38:00

LICA503 Oil level HiLim1 97 11.37.19

* LT100 Obj. desc. LT100 HiLim1 80 95-04-12 11:33:10.068 U
* LT100 Obj. desc. LT100 HiLim2 90 95-04-12 11:33:10.280

Figure 4-28. Default process event list

Choose the event list display from the icons in the upper left-hand corner.

NOTE

The Alarm History is a separate application and is not part of View.

4.4.5.1 Presentation of the last unacknowledged process alarm

The last unacknowledged process alarm is always present in the upper part of the screen.

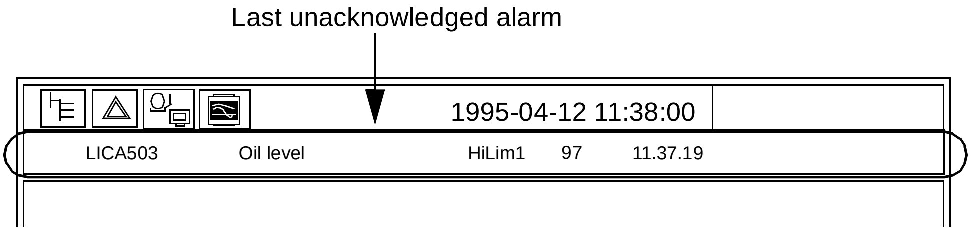


Figure 4-29. Last unacknowledged process alarm

4.4.5.2 Acknowledgment of alarms

You can acknowledge alarms from the following display types:

- Process alarm list.
- Object, group or process displays (select the object producing the alarm).
- Any display from the display header.

For a description, see below.

Acknowledging from lists

All unacknowledged persistent alarms are marked with a flashing red checkmark. Working from the **alarm list** allows you to acknowledge more than one alarm at a time. Acknowledge alarms as follows.

1. Open the **Alarm List**.
2. To acknowledge one alarm:
   a. Select the alarm to acknowledge with the mouse **<Select> key**.
   b. Double-click on the alarm line or press the **<Acknowledge> function key** to acknowledge one alarm.

When there are no more unacknowledged process alarms, the alarm line at the top of the display is cleared.

Acknowledging from object group and process displays

To acknowledge process alarms from other types of displays, do the following.

1. Place the cursor as follows:
    - Object displays: Do not select any cursor placing the object
    - Group displays: On the object name
    - Process displays: On the object symbol or the value displays.
2. Press the **<Acknowledge> key**.

Acknowledging from the display header

1. Place the cursor on the alarm line in the display header.
2. Acknowledge by double-clicking or pressing the **<Acknowledge> function key**.

4.5 Runtime Operation Menus

Refer to the ***View*** manual.

Chapter 5 Maintenance

5.1 Preventive Maintenance

A summary of topics regarding preventive maintenance recommendations and precautions follows. For detailed information, refer to the ***Environment*** manual.

- Precautions when running third party pre-installed hardware and software together with AdvaCommand for Windows.
- When and how to install math co-processors.
- Working with the PC's memory map when installing new hardware.
- About the software key and what to do when it is defective.
- Preventive troubleshooting tips.

5.2 Hardware Indicators

None.

5.3 Error Messages

Refer to the appendix in the ***Alarming and Messaging*** manual for a listing of messages with a description of each for the following topics:

- Start-up messages
- System error messages
- Runtime messages
- Operator-generated messages
- Parts of operator messages.

5.3.1 AdvaSoft FIX interface error messages

FATAL ERROR: Failed to allocate additional IPN entry

Cause: Memory reallocation failed.

Action: Check primary memory in the PC.

Protocol <protocol name> undefined, add it to the PROTOCOLS parameter!

Cause: The protocol not configured in the ADVASOFT.INI file.

Action: Add it to the PROTOCOLS parameter in the COMMUNICATION section.

Error writing to XRF file, file may be corrupt or write-protected

Cause: File writing failed.

Action: Check that file is not write protected.

Invalid object type index: <number>, XRF file probably corrupted

Cause: Failed to add object type.

Action: Try to remove the XRF file.

Failed to allocate array for user defined event treatments

Cause: Memory allocation failed.

Action: Check primary memory in the PC.

Failed to start tt_event subscription for protocol <protocol name>: Status = <error code>

Cause: Alarm and Event subscription towards protocol failed.

Action: Internal error, contact ABB.

bciGetAttribCyclic FAILED bciStatus <error code>

Cause: Cyclic data subscription failed.

Action: Verify that the communication handler is active.

bciExecOperation FAILED bciStatus <error code>

Cause: Object command failed.

Action: Verify that the communication handler is active.

Requested ipn entry not found for <object name>

Cause: Event received for an unknown object. Resolve of object reference failed.

Action: Internal error, contact ABB.

Illegal field = <field number> for ipn = <ipn number>

Cause: Unsupported field name used for object.

Action: Internal error, contact ABB.

Error text: Wrong type in vsp <type number> (vsp2type)

Cause: Illegal object type number found in internal data structure.

Action: Internal error, contact ABB.

Bad resolution in tag name <tag name>

Cause: Time resolution used in tag definition not supported.

Action: Change tag definition from Draw.

Bad unit in tag name <tag name>

Cause: Time unit used in tag definition not supported.

Action: Change tag definition from Draw.

FATAL ERROR: no more protocol slot available!

Cause: Failed to add protocol. Only four protocols supported.

Action: Remove one protocol from the PROTOCOLS parameter in the ADVASOFT.INI file.

XRF file deleted, probably due to new S/W version!

Cause: The ADVACMD.XRF file has been deleted.

Action: Resolve all displays from Draw for better first access display performance.

Error during XRF file deletion!

Cause: Failed to delete the ADVACMD.XRF file.

Action: Check that file is not write protected.

FATAL ERROR: Error allocating memory for object type <type number>

Cause: Memory allocation failed.

Action: Check primary memory in the PC.

FATAL ERROR: Object type <type number> not supported by protocol

Cause: Object type not supported by protocol.

Action: Internal error, contact ABB.

FATAL ERROR: Couldn't add object type <type name>, max of object types reached

Cause: User defined object type could not be added. Only 104 different object types supported.

Action: You can only use 104 different object types.

FATAL ERROR: Directory for XRF file does not exist, check the ADVADIR parameter!

Cause: Could not open the ADVACMD.XRF file.

Action: Check the ADVADIR parameter in the ADVASOFT.INI file and check your hard disk.

5.4 Fault-Finding and User Repair

5.4.1 Troubleshooting

If a problem occurs and originates from within AdvaCommand for Windows software, refer to appendices in the ***Environment*** and ***Alarming and Messaging*** manuals.

Since AdvaCommand for Windows runs together with AdvaSoft for Windows and normally communicates with the automation system via the communication interface, a problem could originate from a number of places. To localize a system or communication error, check the current AdvaSoft for Windows error log and the system status. Refer to the ***AdvaSoft for Windows User's Guide*** and ***Advant Fieldbus 100 Interface User's Guide***.

5.4.1.1 Correcting corrupted historical log files

If the Historical Collect function is interrupted by a power failure, the log files may be corrupted. This can normally be corrected with the "HTC_SCAN" program, found in the FIX directory.

Open the corrupted log file, in the FIX subdirectory HTRDATA\nodename\YYMMDDHH.H04, to correct the file.

5.4.2 How to get support from ABB

If you have questions about, or problems with, AdvaCommand for Windows and you are unable to find or correct the problem through troubleshooting, contact your local ABB representative responsible for this product.

If you encounter errors or if you have suggestions to improve the AdvaSoft for Windows product, please contact your local ABB Industrial Systems AB representative.

When you submit a problem report, it is imperative that you reconstruct the events leading to the problem so we can speedily resolve it. Therefore, please attach any relevant supplementary information, e.g., changes to your system or events prior to the problem. Printouts of System Message logs and hard copies of configuration displays are often crucial.

Appendix A References

A.1 ADVASOFT.INI

Below is the AdvaCommand section, found in the ADVASOFT.INI file.

…
…

```
[AdvaCommand]
SubSeparator=;
```

This line sets the character used as separator between the tagname and the cyclic subscription time setting in AdvaCommand. For example:

```
node:”tag;time”.f_value
```

```
BitSeparator=[
```

This line sets the character used as separator between the attribute and the bitnumber. In the case of the attribute Status:

```
F_Status[0]
```

```
NoAlarmSubs=AF100
```

By giving a protocol name, you can terminate the event and alarm subscription handling for this protocol.

```
SubUsedAttr=AF100
```

This setting is useful for custom made process pictures with one, or a few, attribute subscriptions. Only the requested attributes are subscribed and can, in that way, raise the performance.

…
…

Appendix B References

B.1 HTD.INI

```
[HTD]
Top=65
Left=0
Right=1024
Bottom=768
;
;       ExitKey  - specifies the "quick" exit function key
;                  to exit HTD. Valid input is 2-9 mapping
;                  to F2 to F9, respectively.
ExitKey= 2
[SQL]
;
; This is an example of a historical data request, performed
; against the AS 500IMS. In order to request the Log data,
; you must define the Table names to SQL*Connect. Refer to the
; IMS documentation for further information about the procedure.
; You should change the ##### to specify the actual
; process object name you want to access. The %%%%% should
; specify the actual object attribute (‘MV’, ‘VALUE’, ...) you
; want to access. If you want to access multiple logged objects,
; you may copy this SELECT statement and give it a new name
; (Q2=SELECT....). This name is referred to from your trend
; display pen configuration, together with the DatabaseID
; specified in the ODBC configuration. We recommend you give
; the Query the same name as the process object.
;
Q1=SELECT TIME,Numlogval. NAME,ENTRYVALUE, DESCRIPTION,
LOG_ENG_UNIT,PRES_RANGE_MIN,PRES_RANGE_MAX
FROM Numlogval,Numlogconfig
WHERE (Numlogval.NAME='#####,%%%%')
AND (Numlogconfig.NAME='$HS#####,%%%%-1-o')
ENDQUERY
;
;  Syntax for a Query Statement(this is a valid SQL query):
;    (field order must be as shown)
;
;Q1=SELECT
;D2.TIME,D2.NAME,D2.VAL,T2.DESC,T2.EGU,T2.LEGU,T2.HEGU
;FROM D2,T2
;WHERE (D2.NAME='val1')
;AND (D2.NAME=T2.NAME)
;ENDQUERY
```

```
;
; Where ’Q1’ is the query name (tagsize is the max length (8))
; ’D2’ is the data table name
; ’T2’ is the dictionary table name
; ’TIME’, ’NAME’, ’VAL’,’DESC’,’EGU’,’LEGU’,’HEGU’ are
;    table column names.
; ’val1’ is the variable name stored in the relational db
; ’ENDQUERY’ must follow the last line of the query
;
TTD_AIC71_1=SELECT TIME,Numlogval. NAME,ENTRYVALUE,
Numlogval.NAME,LOG_ENG_UNIT,PRES_RANGE_MIN,PRES_RANGE_MAX
FROM Numlogval,Numlogconfig
WHERE (Numlogval.NAME=’AIC71_1,MV’)
AND (Numlogconfig.NAME=’$HSAIC71_1,MV-1-o’)
ENDQUERY
;
TTD_AIC71_2=SELECT TIME,Numlogval. NAME,ENTRYVALUE,
DESCRIPTION,LOG_ENG_UNIT,PRES_RANGE_MIN,PRES_RANGE_MAX
FROM Numlogval,Numlogconfig
WHERE (Numlogval.NAME=’AIC71_2,MV’)
AND (Numlogconfig.NAME=’$HSAIC71_2,MV-1-o’)
ENDQUERY
```

INDEX

A

AdvaCommand
    program group 1-14
AdvaCommand development 1-2, 1-11
    document binder 1-6
    User's Guides
        Command Language 1-7
        Draw 1-7
        Macro Editor 1-7
        Tag Group Editor 1-7
AdvaCommand runtime 1-2, 1-11
    document binder 1-6
    User's Guides
        Alarming and Messaging 1-7
        DDE Support 1-7
        Environment Setup 1-6
        Historical Trending 1-7
        Security 1-7
        System Configuration 1-6
        View 1-7
Alarm line layout 3-42
Alarm list 4-9, 4-23
    Display font 3-62
    fields in alarm line 3-62
    rearrange column order 3-62
    separate lists for process and system messages 3-55, 3-63
    separate lists for various process sections 3-55, 3-63
Alarm summary
    configuration 3-42
    display 3-55
Alarming 1-12
    report alarms and events 1-2
Alarming and messaging 1-7
Alarms
    acknowledgment 4-26
        from lists 4-26
        from object group and process displays 4-27
        from the display header 4-27
    default alarm line 4-9
    last unacknowledged process alarm 4-26
    unacknowledged 1-12
Alarms and events 4-1
Application building 1-11
AS 500IMS B-1
Assign simple keystroke sequences to macros 1-13
Assigning functions to keystrokes 3-1
Attribute (object) 1-10
Automate operator tasks 1-13

B

Bar graph 1-10
Building tag groups 1-7

C

Calculating capacity and performance 3-1
Capacity 3-2
Chart group 3-64
Color box 1-12
Command language 1-2, 1-13
Common display properties 3-11
Controlling
    analog objects 4-23
    process objects 4-21

D

Data archiving and trending 4-1
DBB 1-2, 1-13
DDE
    for report generation 1-7
Designing alarming and messaging 3-1
Designing displays 3-1
Designing macros 1-7
Designing user access 3-1
Designing user accounts 3-1
Designing write protection 3-1
Developing interactive operator displays 1-7
Dialog area 4-5
Dialog boxes 1-12
Display
    alarm line 4-4
    area 4-4
    areas and layout 3-6
    Basic displays 3-8
    Connect a display to a button 3-43
    Date and Time 3-42, 4-4
    Defining display properties 3-40
    designing group displays 3-52
    direct selection 4-14
    display area layout 4-2
    Display Title 3-41
    Dynamic function key 3-43
    Function key F10 3-44

- group
    - adding paging keys 3-51
- header display area 3-41
- Icons
    - alarm history 4-3
    - alarm list 4-3
    - display menu 4-3
    - historical trend display 4-4
- keyboard function keys 3-44
- linked 4-16
- menu 4-6
    - adding new group displays 3-49
    - choosing displays 4-12
    - creating a group page 3-50
- object displays 3-9, 3-23
- object groups 3-52, 4-7
- paging 4-16
- predefined alarm line layout 3-42
- predefined object displays 4-8
- previous 4-15
- Templates 3-8
- title 4-4

Display elements 1-1
Distributed networking 1-1
Draw 1-12
Dynamic function keys 3-43
Dynamo 1-10
Dynamos 1-2, 1-13
- abbaide 3-23
- abbaode 3-23
- abbdide 3-23
- abbdlg 3-24
- abbdode 3-23
- abbsys 3-24
- AdvaCommand dynamo sets 3-23
- create an object 3-36

E
Error messages 5-1
Event list 4-23, 4-25

F
Fit-to-window 3-12
FIX DataBase Builder 1-2, 1-13
Function keys
- dynamic 4-11 to 4-12

G
Graphic display elements 1-13
Graphical tools 1-2
Graphical user interface 1-12

H
Histogram 1-12
Historical assign 1-7
Historical collect 1-7
Historical data collection, trending and reporting 1-2
Historical log
- correcting corrupted files 5-4

Historical trend display 1-7
Historical trending 1-1, 1-12
HTD.INI A-1, B-1

I
Installation
- AdvaCommand development 2-1
- AdvaCommand runtime 2-1
- installation window 2-1

K
Keyboard keys
- fixed 4-11

Keyboard operation 4-11
KMX file 3-44

M
Macro editor 3-44
Macros 1-2
Macros editor 1-13
Maintenance
- preventive 5-1

Monitor 4-17
Monitor resolution 3-12
- default 3-12, 3-41
- displays used for rescaling 3-12

Monitoring 4-1
- display process data 1-2

Monitoring a process object 4-20
Multi-bar trend 1-12
Multi-pen trend 1-12

N
Networking 1-12
Node 1-11

O
Object attribute

- Analog input 3-25
- Analog output 3-27
- DAT 3-31
- Digital input 3-28
- Digital output 3-30
- reference 3-24

Object closeup 4-5
Object database 3-2
Object display 3-53, 4-20
- assigning displays to existing objects 3-54
- creating a new 3-54

Object group displays 3-52
Object parameters 4-20
Object subscription methods 3-33
- Command scripts 3-33
- Picture refresh rate 3-33
- SQL Command scripts 3-33
- Tag refresh rate 3-33

ODBC driver 3-2
Operating with a mouse 4-12

P

Pen group 3-64
Plant operation 1-11
Predefined and modifiable templates and displays 1-13
Predefined displays 4-1
Predefined template displays 1-1
Process alarms 4-24
Process control
- from object displays 4-22

Process events 4-25
Process object 1-11
Product verification 2-4

R

Real-Time trending 1-12
Real-time trending 1-1
Recommendations
- selecting hardware and software components 1-6

Related documentation 1-5
Release history 1-8
Report
- Excel macros 1-12

Reporting via DDE 1-12
Reports 4-2

S

Screen header 4-3
Security
- configuration 1-7
- user-based 1-12

Selecting a process object 4-18
Setup SQL Database Source 3-2
Sharing of the distributed client/server architecture 1-12
Shutdown procedures 2-3
SQL support 1-12
SQL*Connect B-1
Starting AdvaCommand 4-1
Supervisory control 1-2, 4-1, 4-17
System Configuration Utility 1-6, 2-2
- Setup SQL Database Source 3-2
- SQL Login Account 3-2
- SQL Login information 3-2

T

Tag 1-11
Tag group editor 1-13
Test dynamic data exchange 3-2
Testing displays and data exchange 3-1
Time group 3-64
Tool box 1-12
Trend display 4-10
Troubleshooting 5-4
Tutorial
- QuickStart 3-3

U

User Interface 1-14

V

Variable declarations 3-13
Variables
- add or change 3-13
- global 3-13

View 1-12

X

XY plot 1-12

3BSE 006 237R0201

July, 1997